

**MCS-48™**  
**Microcomputer User's Manual**



## IMPORTANT ERRATA INFORMATION

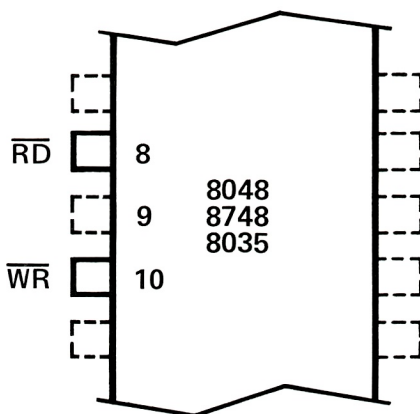
To prevent future inconvenience please take the time to make the following change in pin assignments:

The Read ( $\overline{RD}$ ) and Write ( $\overline{WR}$ ) output pin assignments for the 8048/8748/8035 should be interchanged. Read should be pin 8 and Write should be pin 10.

The following pages are affected:

Pages 2-14,15  
Pages 5-2 through 5-10  
Page 6-1  
Page 6-4

Please make these changes now before you forget. Thank You.



Correct Pinout



# **MCS-48™ MICROCOMPUTER USER'S MANUAL**

This Manual Contains Advance  
Product Information of Which Certain  
Details are Subject to Change

Intellec and MCS are Registered  
Trademarks of Intel Corporation

Copyright© 1976 by Intel Corporation. All rights reserved, no part of this publication including any mnemonics contained herein may be reproduced without the prior written permission of Intel Corporation, 3065 Bowers Avenue, Santa Clara, CA 95051

# TABLE OF CONTENTS

## Chapter 1 Introduction

<b>1.0 Introduction to MCS-48™</b>	1-1
<b>1.1 The Functions of a Computer</b>	1-5
Typical Computer System	1-5
Architecture of a CPU	1-5
Computer Operations	1-8
<b>1.2 Programming a Microcomputer</b>	1-10
Machine Language Programming	1-10
Assembly Language Programming	1-11
<b>1.3 Developing an MCS-48™ Based Product</b>	1-13
Education	1-13
Function Definition	1-13
Hardware Configuration	1-13
Code Generation	1-13
PROMPT 48	1-14
Intellec Development System	1-15
Production	1-15

## Chapter 2 The Single Component MCS-48™ System

<b>2.0 Summary</b>	2-1
<b>2.1 Architecture</b>	2-1
Arithmetic Section	2-1
Program Memory	2-1
Data Memory	2-3
Input/Output	2-4
Test and Interrupt Inputs	2-5
Program Counter and Stack	2-5
Program Status Word	2-6
Conditional Branch Logic	2-6
Interrupt	2-7
Timer Counter	2-8
Clock and Timing Circuits	2-9
Single Step	2-11
Power Down Mode	2-12
External Access Mode	2-13
<b>2.2 Pin Description</b>	2-13
<b>2.3 Programming, Verifying EPROM</b>	2-15
<b>2.4 Test and Debug</b>	2-17
Disabling Internal Program Memory	2-17
Reading Internal Program Memory	2-17

## Chapter 3 The Expanded MCS-48™ System

<b>3.0 Summary</b>	3-1
<b>3.1 Expansion of Program Memory</b>	3-1
Instruction Fetch Cycle	3-1
Extended Addressing	3-1
Restoring Port Information	3-2
Expansion Examples	3-2



## Chapter 3 Continued

3.2 Expansion of Data Memory .....	3-4
Read Write Cycle .....	3-4
Addressing External Memory .....	3-5
Examples of Data Memory Expansion .....	3-5
3.3 Expansion of Input/Output .....	3-5
I/O Expander Device .....	3-6
Expansion with Standard Peripherals .....	3-7
Combination Memory and I/O Expanders .....	3-7
Expansion Examples .....	3-8
3.4 Multi-Chip MCS-48™ Systems .....	3-9
3.5 Bank Switching .....	3-10

## Chapter 4 Instruction Set

4.0 Introduction .....	4-1
Data Transfers .....	4-1
Accumulator Operations .....	4-2
Register Operations .....	4-2
Flags .....	4-2
Branch Instructions .....	4-2
Subroutines .....	4-3
Timer Instructions .....	4-3
Control .....	4-3
Input/Output Instructions .....	4-4
4.1 Instruction Set Description .....	4-4
Summary .....	4-5
Symbols and Abbreviations .....	4-6
Alphabetic Listing by Mnemonic .....	4-7

## Chapter 5 Application Examples

5.0 Introduction .....	5-1
5.1 Hardware Examples .....	5-1
Frequency Reference Options .....	5-1
Stand Alone 8048 .....	5-2
Multiple Interrupts .....	5-3
Prioritized Interrupts .....	5-4
External Program Memory .....	5-5
I/O Expanders .....	5-6
Program Memory and I/O Expander .....	5-8
Data Memory and I/O Expander .....	5-9
Three Chip System .....	5-10
Interface to Drum Printer .....	5-11
Gas Pump Application .....	5-11
Point of Sale Terminal .....	5-12
5.2 Software Examples .....	5-13
Double Arithmetic .....	5-13
Binary Multiply .....	5-14
Interrupt Routine .....	5-15
Byte Processing .....	5-16
A/D Converter .....	5-17

## **Chapter 6 MCS-48™ Component Specifications**

<b>8048</b>	ROM Microcomputer .....	6-1
<b>8748</b>	EPROM Microcomputers .....	6-1
<b>8035</b>	Microcomputers .....	6-1
<b>8355</b>	ROM and I/O Expander .....	6-7
<b>8755</b>	EPROM and I/O Expander .....	6-13
<b>8155</b>	RAM and I/O Expander .....	6-19
<b>8243</b>	MCS-48™ I/O Expander .....	6-29

## **Chapter 7 Compatible MCS-80™ Components**

<b>8308</b>	8192 Bit Static MOS ROM .....	7-1
<b>8316A</b>	16,384 Bit Static MOS ROM .....	7-5
<b>8708</b>	8192 1K x 8 EPROM .....	7-11
<b>8101A-4</b>	1024 Bit Static MOS RAM With Separate I/O .....	7-15
<b>8111A-4</b>	1024 Bit Static MOS RAM With Common I/O .....	7-19
<b>5101</b>	1024 Bit Static CMOS RAM .....	7-23
<b>8212</b>	Eight-Bit Input/Output Port .....	7-27
<b>8255A</b>	Programmable Peripheral Interface .....	7-37
<b>8251</b>	Programmable Communication Interface .....	7-59
<b>8205</b>	High Speed 1 Out of 8 Binary Decoder .....	7-73
<b>8214</b>	Priority Interrupt Control Unit .....	7-79
<b>8216/8226</b>	4-Bit Parallel Bi-Directional Bus Driver .....	7-83
<b>8253</b>	Programmable Interval Timer .....	7-89
<b>8259</b>	Programmable Interrupt Controller .....	7-101
<b>8279</b>	Programmable Peripheral Interface .....	7-117

## **Appendices**

<b>Packaging Information</b> .....	A1-1
<b>Ordering Information</b> .....	A2-1



# Chapter 1

## INTRODUCTION



# INTRODUCTION

1.0 Introduction to MCS-48™ .....	1-1
1.1 Functions of a Computer .....	1-5
1.2 Programming a Microcomputer .....	1-10
1.3 Developing An MCS-48™ Based Product .....	1-13



# INTRODUCTION

## 1.0 Introduction to MCS-48™

Recent advances in NMOS technology have allowed Intel for the first time to place enough capability on a single silicon die to create a true single-chip microcomputer containing all the functions required in a digital processing system. This microcomputer, its variations, and its optional peripherals are collectively called the MCS-48 microcomputer family and are fully described in this manual.

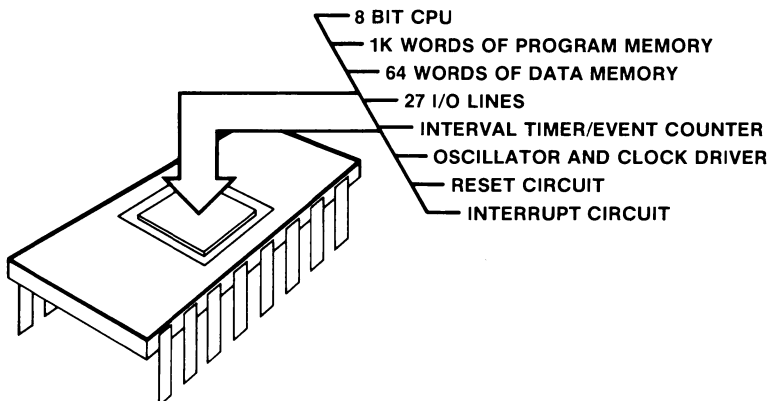
The head of the family is the 8048 microcomputer which contains the following functions in a single 40 pin package:

- 8-Bit CPU
- 1K x 8 ROM Program Memory
- 64 x 8 RAM Data Memory
- 27 I/O Lines
- 8-Bit Timer/Event Counter

A 2.5 or 5.0 microsecond cycle time and a repertoire of over 90 instructions each consisting of either one or two cycles makes

the single chip 8048 the equal in performance of most presently available multi-chip NMOS microprocessors, yet the 8048 is a true "low-cost" microcomputer. A single 5V supply requirement for all MCS-48 components assures that "low cost" also applies to the power supply in your system.

Even with low component costs; however, a project may be jeopardized by high development and rework costs resulting from an inflexible production design. Intel has solved this problem by creating two pin-compatible versions of the 8048 microcomputer: the 8048 with mask Programmable ROM program memory for low cost production and the 8748 with user programmable and erasable EPROM program memory for prototype development. The 8748 is essentially a single chip microcomputer "bread-board" which can be modified over and over again during development and pre-production then simply replaced by the low cost 8048 ROM for volume production. The 8748



provides a very easy transition from development to production and also provides an easy vehicle for temporary field updates while new ROMs are being made.

---

### SPECIAL FEATURES

- **SINGLE 5V SUPPLY**
  - **40 PIN DIP**
  - **PIN COMPATIBLE ROM AND EPROM**
  - **2.5 and 5.0  $\mu$ sec CYCLE VERSIONS**
  - **ALL INSTRUCTIONS 1 OR 2 CYCLES**
  - **SINGLE STEP**
  - **8 LEVEL STACK**
  - **2 WORKING REGISTER BANKS**
  - **RC, XTAL, OR EXTERNAL FREQUENCY SOURCE**
  - **CLOCK PER CYCLE AND OPTIONAL CLOCK PER STATE OUTPUT**
- 

To allow the MCS-48 to solve a wide range of problems and to provide for future expansion, all 8048 functions have been made externally expandable using either special expanders or standard memories and peripherals. An efficient low cost means of I/O expansion is provided by the 8243 I/O Expander which provides 16 I/O lines in a 24 pin package. For systems with large I/O requirements multiple 8243s can be used.

For such applications as Keyboards, Displays, Serial communication lines, etc. standard MCS-80™ (8080) peripheral circuits may be added. Program and data memory may be expanded using standard memories or the 8355 and 8155 memories that also include programmable I/O lines and timing functions.

The 8035 is an 8048 without internal program memory that allows the user to match his program memory requirements exactly by using a wide variety of external memories. The 8035 allows the user to select a minimum cost system no matter what his program memory requirements.

The 8048 was designed to be an efficient control processor as well as an arithmetic processor with an instruction set which allows the user to directly set and reset individual lines within its I/O ports as well as test individual bits within the accumulator. A large variety of branch and table look-up instructions make the 8048 very efficient in implementing standard logic functions. Special attention was also given to code efficiency with over 70% of the instructions being single byte and all others being only two bytes. This means many functions requiring 1.5K to 2.0K bytes in other processors may very well be compressed into the 1K words resident in the 8048.

---

### THE MCS-48™ FAMILY

**8048 — MICROCOMPUTER WITH ROM**

**8748 — MICROCOMPUTER WITH EPROM**

**8035 — MICROCOMPUTER WITHOUT ROM**

**8243 — I/O EXPANDER**

**8355 — ROM PROGRAM MEMORY AND I/O EXPANDER**

**8755 — EPROM PROGRAM MEMORY AND I/O EXPANDER**

**8155 — DATA MEMORY AND I/O EXPANDER**

---

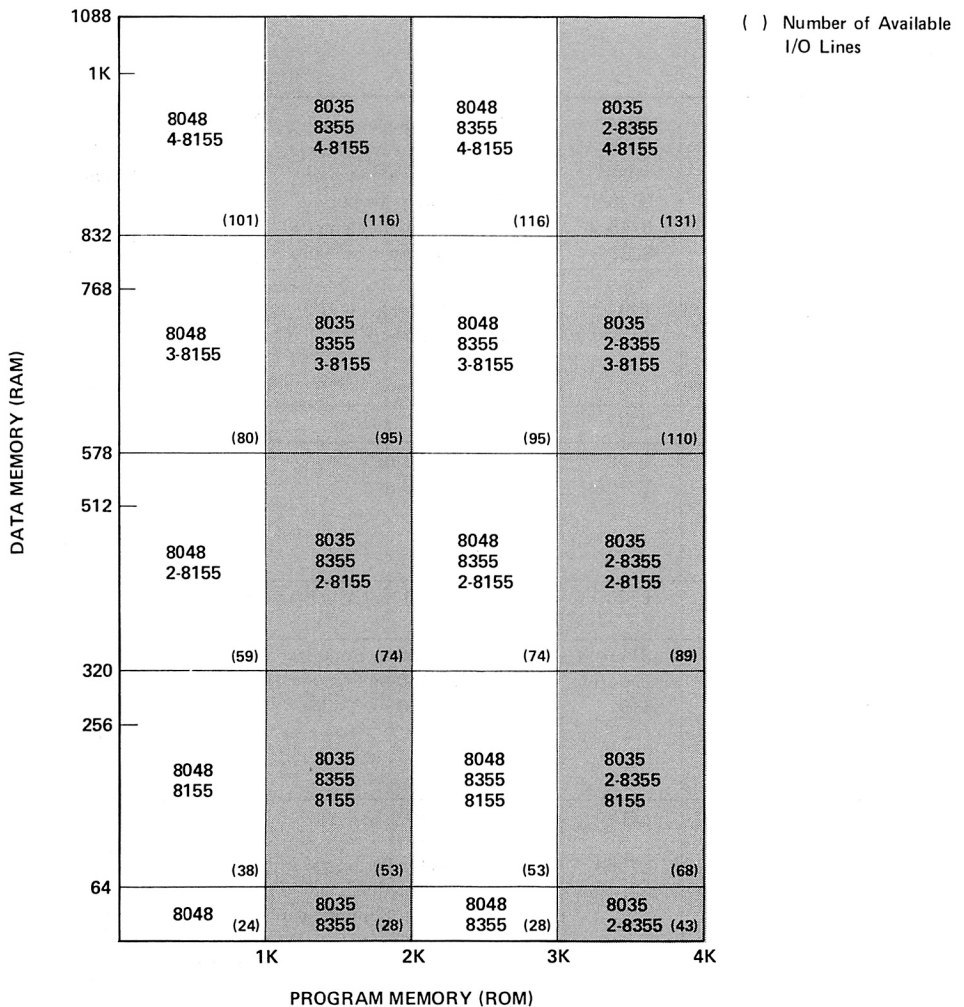


# INTRODUCTION

MCS-48	Microcomputers	8048 8748 8035 8048-8 8748-8 8035-8	<div> <div>ROM Program Memory</div> <div>EPROM Program Memory</div> <div>External Program Memory</div> </div> <div> <div>ROM Program Memory</div> <div>EPROM Program Memory</div> <div>External Program Memory</div> </div>	<div>2.5 <math>\mu</math>sec Cycle</div> <div>5.0 <math>\mu</math>sec Cycle</div>	Three compatible versions of the single chip micro-computers provide mask programmed, light erasable, or no internal program memory.
	Memory and I/O Expanders	8355 8755 8155	2K x 8 ROM with 16 I/O Lines 2K x 8 EPROM with 16 I/O Lines 256 x 8 RAM with 22 I/O Lines and Timer		Compatible devices allow direct expansion of 8048/8748/8035 functions with no additional external components.
	I/O Expander	8243	16 Line I/O Expander		Low Cost I/O Expander
Compatible MCS-80™ Components	Standard ROMs	8308 8316A	1K x 8 450 ns 2K x 8 850 ns		Allow low cost external expansion of Program Memory. The 8308 is interchangeable with 8708.
	Standard EPROM	8708	1K x 8 450 ns Light Erasable		User programmable and erasable.
	Standard RAMs	8111A-4 8101A-4 5101	256 x 4 450 ns Common I/O 256 x 4 450 ns Separate I/O 256 x 4 650 ns CMOS		Data memory can be easily expanded using standard NMOS RAMs. The 5101 CMOS equivalent reduces standby power to 75 nW/bit
	Standard I/O	8212  8255A  8251	8-Bit I/O Port  Programmable Peripheral Interface  Programmable Communicating Interface		Serves as Address Latch or I/O port. Three 8-bit programmable I/O ports. Serial Communications Receiver/ Transmitter
	Standard Peripherals	8205 8214 8216 8226 8253 8259 8279	1 of 8 Binary Decoder Priority Interrupt Controller Bi-directional Bus Driver Bi-directional Bus Driver (Inverting) Programmable Interval Timer Programmable Interrupt Controller Programmable Keyboard/Display Interface		MCS-80 peripheral devices are compatible with the MCS-48 allowing easy addition of such specialized interfaces as the 8279 Keyboard/Display Interface. Future MCS-80 devices will also be compatible.

## MCS-48™ MICROCOMPUTER COMPONENTS

# INTRODUCTION



## THE EXPANDED MCS-48™ SYSTEM

The chart above shows the various expansion possibilities using the 8048/8748 or the 8035 in various combinations with the 8355/8755 Program Memory and I/O Expander and the 8155 Data Memory and I/O Expander. Data Memory can be expanded beyond the resident 64 words in blocks of 256 by adding 8155's. Program Memory can be expanded beyond the resident 1K in blocks

of 1K by using the 8355/8755 in combination with the 8035 or 8048. Since the 8355 contains 2K words the 8035 is needed to fill in the "gaps". For program memory of 1K or less use the 8048. For programing in the 1 to 2K range use an 8035/8355 combination and for the 2 to 3K range use an 8048/8355 combination.

## 1.1 The Functions of a Computer

This chapter introduces certain basic computer concepts. It provides background information and definitions which will be useful in later chapters of this manual. Those already familiar with computers may skip this material, at their option.

### 1.1.1 A Typical Computer System

A typical digital computer consists of:

- A central processor unit (CPU)
- Program Memory
- Data Memory
- Input/output (I/O) ports

The processor memory serves as a place to store Instructions, the coded pieces of information that direct the activities of the CPU, while Memory stores the Data, the coded pieces of information that are processed by the CPU. A group of logically related instructions stored in memory is referred to as a Program. The CPU "reads" each instruction from memory in a logically determined sequence, and uses it to initiate processing actions. If the program sequence is coherent and logical, processing the program will produce intelligible and useful results. The program must be organized such that the CPU does not read a non-instruction word when it expects to see an instruction.

The CPU can rapidly access any data stored in memory; but often the memory is not large enough to store the entire data bank required for a particular application. The problem can be resolved by providing the computer with one or more Input Ports. The CPU can address these ports and input the data contained there. The addition of input ports enables the computer to receive information from external equipment (such as a paper tape reader or floppy disk) at high rates of speed and in large volumes.

A computer also requires one or more Output Ports that permit the CPU to communicate the result of its processing to the outside world. The output may go to a display, for use by a human operator, to a peripheral device that produces "hard-copy", such as a line-

printer, to a peripheral storage device, such as a floppy disk unit, or the output may constitute process control signals that direct the operations of another system, such as an automated assembly line. Like input ports, output ports are addressable. The input and output ports together permit the processor to communicate with the outside world.

The CPU unifies the system. It controls the functions performed by the other components. The CPU must be able to fetch instructions from memory, decode their binary contents and execute them. It must also be able to reference memory and I/O ports as necessary in the execution of instructions. In addition, the CPU should be able to recognize and respond to certain external control signals, such as INTERRUPT requests. The functional units within a CPU that enable it to perform these functions are described below.

### 1.1.2 The Architecture of a CPU

A typical central processor unit (CPU) consists of the following interconnected functional units:

- Registers
- Arithmetic/Logic Unit (ALU)
- Control Circuitry

Registers are temporary storage units within the CPU. Some registers, such as the program counter and instruction register, have dedicated uses. Other registers, such as the accumulator, are for more general purpose use.

#### Accumulator

The accumulator usually stores one of the operands to be manipulated by the ALU. A typical instruction might direct the ALU to add the contents of some other register to the contents of the accumulator and store the result in the accumulator itself. In general, the accumulator is both a source (operand) and a destination (result) register. Often a CPU will include a number of additional general purpose registers that can be used to store operands or intermediate data. The availability of general purpose registers

eliminates the need to “shuffle” intermediate results back and forth between memory and the accumulator, thus improving processing speed and efficiency.

### **Program Counter (Jumps, Subroutines and the Stack):**

The instructions that make up a program are stored in the system's memory. The central processor references the contents of memory in order to determine what action is appropriate. This means that the processor must know which location contains the next instruction.

Each of the locations in memory is numbered, to distinguish it from all other locations in memory. The number which identifies a memory location is called its Address. The processor maintains a counter which contains the address of the next program instruction. This register is called the Program Counter. The processor updates the program counter by adding “1” to the counter each time it fetches an instruction, so that the program counter is always current (pointing to the next instruction).

The programmer therefore stores his instructions in numerically adjacent addresses, so that the lower addresses contain the first instructions to be executed and the higher addresses contain later instructions. The only time the programmer may violate this sequential rule is when an instruction in one section of memory is a Jump instruction to another section of memory.

A jump instruction contains the address of the instruction which is to follow it. The next instruction may be stored in any memory location, as long as the programmed jump specifies the correct address. During the execution of a jump instruction, the processor replaces the contents of its program counter with the address embodied in the Jump. Thus, the logical continuity of the program is maintained.

A special kind of program jump occurs when the stored program “Calls” a subroutine. In

this kind of jump, the processor is required to “remember” the contents of the program counter at the time that the jump occurs. This enables the processor to resume execution of the main program when it is finished with the last instruction of the subroutine.

A Subroutine is a program within a program. Usually it is a general-purpose set of instructions that must be executed repeatedly in the course of a main program. Routines which calculate the square, the sine, or the logarithm of a program variable are good examples of functions often written as subroutines. Other examples might be programs designed for inputting data to a particular peripheral device.

The processor has a special way of handling subroutines, in order to insure an orderly return to the main program. When the processor receives a Call instruction, it increments the Program Counter and stores the counter's contents in a reserved memory area known as the Stack. The Stack thus saves the address of the instruction to be executed after the subroutine is completed. Then the processor loads the address specified in the Call into its Program Counter. The next instruction fetched will therefore be the first step of the subroutine.

The last instruction in any subroutine is a Return. Such an instruction need specify no address. When the processor fetches a Return instruction, it simply replaces the current contents of the Program Counter with the address on the top of the stack. This causes the processor to resume execution of the calling program at the point immediately following the original Call instruction.

Subroutines are often Nested; that is, one subroutine will sometimes call a second subroutine. The second may call a third, and so on. This is perfectly acceptable, as long as the processor has enough capacity to store the necessary return addresses, and the logical provision for doing so. In other words, the maximum depth of nesting is determined by the depth of the stack itself. If the stack has space for storing three return addresses, then



three levels of subroutines may be accommodated.

### **Instruction Register and Decoder**

Every computer has a Word Length that is characteristic of that machine. A computer's word length is usually determined by the size of its internal storage elements and interconnecting paths (referred to as Buses); for example, a computer whose registers and buses can store and transfer 8-bits of information has a characteristic word length of 8-bits and is referred to as an 8-bit parallel processor. An 8-bit parallel processor generally finds it most efficient to deal with 8-bit binary fields, and the memory associated with such a processor is therefore organized to store 8-bits in each addressable memory location. Data and instructions are stored in memory as 8-bit binary numbers, or as numbers that are integral multiples of 8-bits: 16-bits, 24-bits, and so on. This characteristic 8-bit field is often referred to as a Byte. If however, efficient handling of 4 or even 1-bit data is necessary special processor instructions can provide this capability.

Each operation that the processor can perform is identified by a unique byte of data known as an Instruction Code or Operation Code. An 8-bit word used as an instruction code can distinguish between 256 alternative actions, more than adequate for most processors.

The processor fetches an instruction in two distinct operations. First, the processor transmits the address in its Program Counter to the program memory. Then the program memory returns the addressed byte to the processor. The CPU stores this instruction byte in a register known as the Instruction Register, and uses it to direct activities during the remainder of the instruction execution.

The 8-bits stored in the instruction register can be decoded and used to selectively activate one of a number of output lines. Each line represents a set of activities associated with execution of a particular instruction code. The enabled line can be combined with selected timing pulses, to develop electrical

signals that can then be used to initiate specific actions. This translation of code into action is performed by the Instruction Decoder and by the associated control circuitry.

An 8-bit instruction code is often sufficient to specify a particular processing action. There are times, however, when execution of the instruction requires more information than 8-bits can convey.

One example of this is when the instruction references a memory location. The basic instruction code identifies the operation to be performed, but cannot specify the object address as well. In a case like this, a two byte instruction must be used. Successive instruction bytes are stored in sequentially adjacent memory locations, and the processor performs two fetches in succession to obtain the full instruction. The first byte retrieved from memory is placed in the processor's instruction register, and subsequent byte is placed in temporary storage; the processor then proceeds with the execution phase.

### **Address Register(s)**

A CPU may use a register to hold the address of a memory location that is to be accessed for data. If the address register is Programmable, (i.e., if there are instructions that allow the programmer to alter the contents of the register) the program can "build" an address in the address register prior to executing a Memory Reference instruction (i.e., an instruction that reads data from memory, writes data to memory or operates on data stored in memory).

### **Arithmetic/Logic Unit (ALU)**

All processors contain an arithmetic/logic unit, which is often referred to simply as the ALU. The ALU, as its name implies, is that portion of the CPU hardware which performs the arithmetic and logical operations on the binary data.

The ALU must contain an Adder which is capable of combining the contents of two registers in accordance with the logic of binary arithmetic. This provision permits the

processor to perform arithmetic manipulations on the data it obtains from memory and from its other inputs.

Using only the basic adder a capable programmer can write routines which will subtract, multiply and divide, giving the machine complete arithmetic capabilities. In practice, however, most ALUs provide other built-in functions, including boolean logic operations, and shift capabilities.

The ALU contains Flag Bits which specify certain conditions that arise in the course of arithmetic and logical manipulations. It is possible to program jumps which are conditionally dependent on the status of one or more flags. Thus, for example, the program may be designed to jump to a special routine if the carry bit is set following an additional instruction.

### Control Circuitry

The control circuitry is the primary functional unit within a CPU. Using clock inputs, the control circuitry maintains the proper sequence of events required for any processing task. After an instruction is fetched and decoded, the control circuitry issues the appropriate signals (to units both internal and external to the CPU) for initiating the proper processing action. Often the control circuitry will be capable of responding to external signals, such as an interrupt. An Interrupt request will cause the control circuitry to temporarily interrupt main program execution, jump to a special routine to service the interrupting device, then automatically return to the main program.

#### 1.1.3 Computer Operations

There are certain operations that are basic to almost any computer. A sound understanding of these basic operations is a necessary prerequisite to examining the specific operations of a particular computer.

### Timing

The activities of the central processor are cyclical. The processor fetches an instruction, performs the operations required,

fetches the next instruction, and so on. This orderly sequence of events requires precise timing, and the CPU therefore requires a free running oscillator clock which furnishes the reference for all processor actions. The combined fetch and execution of a single instruction is referred to as an Instruction Cycle. The portion of a cycle identified with a clearly defined activity is called a State. And the interval between pulses of the timing oscillator is referred to as a Clock Period. As a general rule, one or more clock periods are necessary for the completion of a state, and there are several states in a cycle.

### Instruction Fetch

The first state(s) of any instruction cycle will be dedicated to fetching the next instruction. The CPU issues a read signal and the contents of the program counter are sent to program memory, which responds by returning the next instruction word. The first byte of the instruction is placed in the instruction register. If the instruction consists of more than one byte, additional states are required to fetch the second byte of the instruction. When the entire instruction is present in the CPU, the program counter is incremented (in preparation for the next instruction fetch) and the instruction is decoded. The operation specified in the instruction will be executed in the remaining states of the instruction cycle. The instruction may call for a data memory read or write, an input or output and/or an internal CPU operation, such as a register-to-register transfer or an add operation.

### Memory Read

An instruction fetch is merely a special program memory read operation that brings the instruction to the CPU's instruction register. The instruction fetched may then call for data to be read from data memory into the CPU. The CPU again issues a read signal and sends the proper memory address; memory responds by returning the requested word. The data received is placed in the accumulator or one of the other general purpose registers (not the instruction register).

### Memory Write

A memory write operation is similar to a read except for the direction of data flow. The CPU issues a write signal, sends the proper memory address, then sends the data word to be written into the addressed data memory location.

### Input/Output

Input and Output operations are similar to memory read and write operations with the exception that an I/O port is addressed instead of a memory location. The CPU issues the appropriate input or output control signal, sends the proper address and either receives the data being input or sends the data to be output.

Data can be input/output in either parallel or serial form. All data within a digital computer is represented in binary coded form. A binary data word consists of a group of bits; each bit is either a one or a zero. Parallel I/O consists of transferring all bits in the word at the same time, one bit per line. Serial I/O consists of transferring one bit at a time on a single line. Naturally serial I/O is much slower, but it requires considerably less hardware than does parallel I/O.

### Interrupts

Interrupt provisions are included on many central processors, as a means of improving

the processor's efficiency. Consider the case of a computer that is processing a large volume of data, portions of which are to be output to a printer. The CPU can output a byte of data within a single machine cycle but it may take the printer the equivalent of many machine cycles to actually print the character specified by the data byte. The CPU could then remain idle waiting until the printer can accept the next data byte. If an interrupt capability is implemented on the computer, the CPU can output a data byte then return to data processing. When the printer is ready to accept the next data byte, it can request an interrupt. When the CPU acknowledges the interrupt, it suspends main program execution and automatically branches to a routine that will output the next data byte. After the byte is output, the CPU continues with main program execution. Note that this is, in principle, quite similar to a subroutine call, except that the jump is initiated externally rather than by the program.

More complex interrupt structures are possible, in which several interrupting devices share the same processor but have different priority levels. Interruptive processing is an important feature that enables maximum utilization of a processor's capacity for high system throughput.

## 1.2 Programming a Microcomputer

### 1.2.1 Machine Language Programming

A microprocessor is instructed what to do by programming it with a series of instructions stored in Program Memory. The processor fetches these instructions one at a time and performs the operation indicated. These instructions must be stored in a form that the processor can understand. This format is referred to as Machine Language. For most microprocessors this instruction is a group of 8 binary bits (1's and 0's) called a word (also called a byte if the word is 8-bits). Some instructions require more than one location in Program Memory. To execute a multi-byte instruction, the processor must execute multiple fetches of program memory before performing the instruction. Because multi-byte instructions take more Program Memory and take longer to execute than single byte instructions their use is usually kept to a minimum.

A processor may be programmed by writing a sequence of instructions in the binary code (ones and zeros) which the machine can interpret directly. This is machine language programming and it is very useful where the program to be written is small and the application requires that the designer have an intimate knowledge of the microprocessor. Machine language programming allows the user, because of his detailed knowledge, to use many programming "tricks" to produce the most compact and efficient code possible.

The following is an example of a machine language program: This program reads 5 sequential 8-bit words in from an I/O port and stores them sequentially in data memory. The program starts by initializing two registers, one which determines where the data is to be stored and another which

counts the number of words to be stored. When finished the processor continues on to the next instructions.

Step Number	Machine Code	Explanation
0	1011 1000	Load decimal 32 in register R0
1	0010 0000	
2	1011 1010	Load decimal 5 in register R2
3	0000 0101	
4	0000 1001	Load Port 1 to accumulator
5	1111 0000	Transfer contents of accumulator to register addressed by register 0
6	0001 1000	Increment R0 by 1
7	1110 1010	Decrement register 2 by 1, if result is zero continue to step 9, if not go to step 4
8	0000 0100	
9	—	
10	—	

As you can see, writing machine instructions in ones and zeros can be very laborious and subject to error. It is almost always more efficient to represent each 8-bits of machine language code in a shorthand format called Hexadecimal. The term hexadecimal results from the character set used in hexadecimal notation. Hexadecimal is merely an extension of the normal decimal numbers by the addition of the first six letters of the alphabet. This gives a total of 16 different characters. Each hexadecimal "digit" can represent 16 values or the equivalent of four binary bits; therefore, each 8-bit machine language word can be represented by 2 hexadecimal (hex for short) digits. The correspondence among the decimal, binary, and hex number systems is given below:

Decimal	Hex	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Our machine language program then becomes:

Step	Hex Code
0	B8
1	20
2	BA
3	05
4	09
5	F0
6	18
7	EA
8	04

This coding is now quite efficient to write and read and coding errors are much easier to detect. Hex coding is usually very efficient for small programs (a few hundred lines of code) however, it does have two major limitations in larger programs:

1. Hex coding is not self-documenting, that is, the code itself does not give any indication in human terms of the operation to be performed. The user must learn each code or constantly use a Program Reference Card to convert.
2. Hex coding is absolute, that is, the program will work only when stored in a specific location in program memory. This is because the branch or jump instructions in the program reference specific addresses elsewhere in the program. In the example above steps 7 and 8 reference step (or address) 4. If the program were to be moved,

step 8 would have to be changed to refer to the new address of step 4.

## 1.2.2 Assembly Language Programming

Assembly language overcomes the disadvantages of machine language by allowing the use of alphanumeric symbols to represent machine operation codes, branch addresses, and other operands. For example, the instruction to increment the contents of register 0 becomes INC R0 instead of the hex 18, giving the user at a glance the meaning of the instruction. Our example program can be written in assembly language as follows:

Step No.	Hex Code	Assembly Code
0	B8	MOV R0, #32
1	20	
2	BA	MOV R2, #05
3	05	
4	09	INP: IN A, P1
5	F0	MOV @R0, A
6	18	INC R0
7	EA	DJNZ R2, INP
8	04	

The first statement can be verbalized as follows: Move to Register 0 the decimal number 32. Move instructions are always structured such that the destination is first and the source is second. The pound sign “#” indicates that the source is “immediate” data (data contained in the following byte of program memory). In this case data was specified as a decimal 32, however, this could have been written as a hex 20H or a binary 0010 0000B since the assembler will accept either form. Notice also that in this instance two lines of hex code are represented by one line of assembly code.

The input instruction IN A, P1 has the same form as a MOV instruction indicating that the contents of Port 1 are to be transferred to the accumulator. In front of the input instruction is an address label which is delineated by a colon. This label allows the program to be written in a form independent of its final location in program memory since the branch instruction at the end of the program can refer to this label rather than a specific address. This is a very important advantage of assembly language programs since it

allows instructions to be added or deleted throughout the program during debugging without requiring that any jump addresses be changed.

The next instruction `MOV @R0, A` can be verbalized as, Move to the data memory location addressed by R0, the contents of the accumulator. The @ sign indicates an indirect operation whereby the contents of either register 0 or register 1 acts as a pointer to the data memory location to be operated on.

The last instruction is a Decrement and Jump if Not Zero instruction which acts in combination with the specified register as a loop counter. In this case register 2 is loaded with 5 initially and then decremented by one each time the loop is executed. If the result of the decrement is not zero, the program jumps to INP and executes another input operation. The fifth time thru the loop the result is zero and execution falls through to whatever routine follows the DJNZ instruction.

In addition to the normal features provided by assemblers, more advanced assemblers such as that for the MCS-48 offer such things as evaluation of expressions at assembly time, conditional assembly, and macro capability.

1. Evaluation of Expressions - Certain assemblers allow the use of arithmetic expressions and multiple symbols in the operand portion of instructions. For instance the MCS-48 assembler accepts instructions such as:

```
ADD A, # ALFA*BETA/2
```

ALFA and BETA are two previously defined symbols. At assembly time the expression `ALFA*BETA/2` will be evaluated and the resulting number (which is the average of ALFA and BETA) will be treated as immediate data and designated as the second byte of the ADD immediate instruction. This expression has allowed the immediate data of this instruction to be defined in a single statement and eliminated the need for a third symbol equal to `ALFA*BETA/2`.

2. Conditional Assembly - Conditional assembly allows the programmer to select only certain portions of his assembly language (source) program for conversion to machine (object) code at assembly time. This allows for instance, the inclusion of various "debug" routines to be included in the program during development. Using conditional assembly, they can then be left out when the final assembly is done.

Conditional assembly also allows several versions of one basic program to be generated by selecting various portions of a larger program at assembly time.

3. Macro's - A macro instruction is essentially a symbol which is recognized by the assembler to represent a specific sequence of several standard instructions. A macro is a shorthand way of generating the same sequence of instructions at several locations in a program without having to rewrite the sequence each time it is used. For example, a typical macro instruction might be one which performs a subtract operation. The 8048 does not have a subtract instruction as such but the operation can be performed easily with three instructions:

```
CPL A
ADD A, REG
CPL A
```

This routine subtracts a register from the accumulator and leaves the result in the accumulator. This sequence can be defined as a macro with the name SUB and an operand which can be R0 to R7. To subtract R7 from the accumulator then, the programmer merely has to write:

```
SUB R7
```

and the assembler will automatically insert the three instructions above with R7 substituted for REG.

Once the assembly language source code is written it can be converted to machine executable object code by passing it through an assembler program. The MCS-48 assembler is a program which runs on the 8080-based Intellec MDS system explained in the next section.

### 1.3 Developing An MCS-48™ Based Product

Although the development of a microcomputer based product may differ in detail from the development cycle of a product based on TTL logic or relays, the basic procedures are the same — only the tools are different.

#### 1.3.1 Education

The first step of course is to become familiar with what the microcomputer is and what it can do. The first step in this education is this document, the MCS-48™ User's Manual. The user's manual gives a detailed description of the MCS-48 family of components and how they may be used in various system configurations. Also included is a description of the 8048 instruction set and examples of how the instructions may be used. For a more complete discussion of the instruction set and programming techniques the MCS-48 Assembly Language Manual is also available.

If time is critical in getting started in microcomputers, individuals can attend one of many Intel sponsored 3-day training courses which give basic instruction in the MCS-48 as well as hands-on experience with MCS-48 development systems. These courses are a convenient means of getting started with the MCS-48, particularly for those not familiar with microprocessors.

After general familiarization is complete, either through self-instruction or a training course, the next step is to gain a better "feel" for what a microprocessor can do in your own applications by writing several exercise programs which perform basic functions. You may require such things as I/O routines, delays, counting functions, look-up tables, arithmetic functions, and logical operations which can serve as a set of building blocks for future applications programs. Several basic programming examples are included in the MCS-48 Assembly Language Manual while the Intel User's Library is a source of more specific applications routines.

#### 1.3.2 Function Definition

After a thorough understanding of the

microprocessor is achieved, the functions to be implemented can be defined using a flowchart method to describe each basic system function and the sequence in which the processor executes these functions. Once the system is flowcharted, critical time-related functions can be identified and sample programs written to verify that performance requirements can be met.

#### 1.3.3 Hardware Configuration

The next step involves the definition of the microcomputer hardware required to implement the function. Input/Output capability must be defined in terms of number of inputs, number of outputs, bi-directional lines, latching or non-latching I/O, output drive capability, and input impedance. The number of words of RAM storage required for intermediate results and data storage must then be determined. The type of system will dictate whether battery backup is needed to maintain data RAM during power failure.

Probably the most difficult parameter to define initially is the amount of program memory needed to store the applications program. Although previously written exercise programs will make this estimate more accurate, a generous amount of "breathing room" should be allowed in program memory until coding is complete and the exact requirements are known. Many special functions such as serial communications (TTY) or keyboard/display interfaces may be implemented in software (programs); however, in cases where these functions place a severe load on the processor in terms of time or program memory, special peripheral interface circuits such as the 8251, Universal Synchronous or Asynchronous Receiver/Transmitter (USART) or 8279 Keyboard/Display interface may be used.

#### 1.3.4 Code Generation

The writing of the final program code for the application can begin once the system function and hardware have been defined and can be generated in parallel with the detailed hardware design (PC card layout, power supply, etc.)

At this point, there are two paths available to the designer/programmer and two types of design development aids provided by Intel to simplify the procedures. One system, called PROMPT 48, is a low cost development system which supports machine language programming and the second is the Intel Microcomputer Development System which supports both machine and assembly languages. For those of you unfamiliar with the advantages and disadvantages of machine and assembly languages see Section 1.2.

## 1.3.5 PROMPT 48

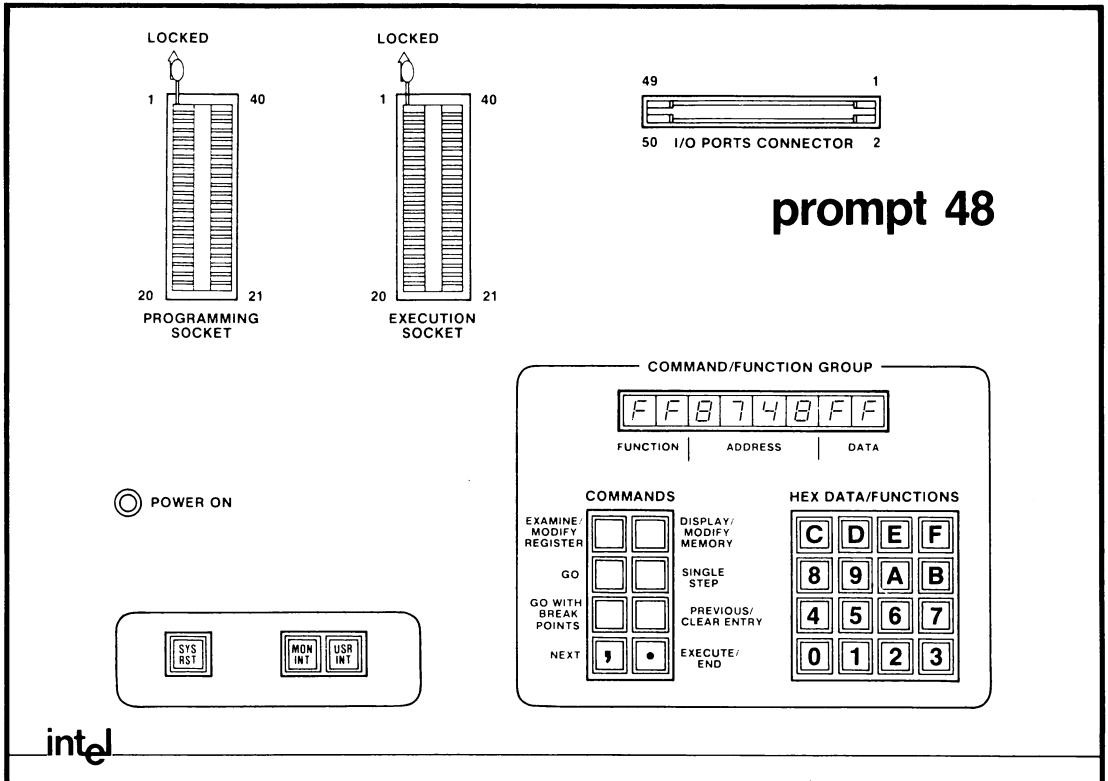
PROMPT 48 is a low cost design aid consisting of: an 8748 processor to execute programs, control circuitry to provide debug functions such as single step and break points, a monitor program stored in ROM, an EPROM programmer, and a hexadecimal keyboard and display. There are two processor sockets on the front of PROMPT 48, one for programming the 8748 and one in

which a programmed 8748 executes its program while under control of the monitor routine.

Use of PROMPT 48 involves the following steps:

1. Loading an application program into the PROMPT RAM memory via Hex keyboard or external terminal (TTY and RS232 interface provided).
2. Inserting an erased 8748 in the programming socket and transferring the application program to its internal EPROM.
3. Transferring programmed 8748 to execution socket where program is executed and debugged under control of the monitor.

The monitor routine allows the user to single step this processor, examine or modify all internal registers and data memory; or to run at full speed and stop the processor at predetermined breakpoints. PROMPT 48





also provides 1K of writeable program memory which may be used to debug user programs. A multiple single step feature is also provided in which the processor steps through its program dumping all internal contents to external RAM where it may be later displayed or typed out on an external terminal. Paper tape input and output in Intel's hexadecimal format is also available through the TTY.

### 1.3.6 Intellec Development System

The Intellec Microcomputer Development System is a modular development system which can be expanded as necessary to meet the requirements of your design cycle. The system consists of the processor unit which is based on Intel's 8080A microprocessor, and several optional units such as the UPP Universal PROM Programmer, the PTR High Speed Paper tape reader, the DOS Disk Operating System, and the Intellec CRT terminal.

To support the development of MCS-48 systems a macro-assembler ASM 48 is available for the Intellec System as well as a personality module for the UPP which will program the EPROM of the 8748. Also to be provided is in-circuit emulation capability with ICE-48 which will allow emulation and debug of user's 8048 application programs on the 8080A-based Intellec Development System.

The Intellec system is a flexible high performance development system which can support Intel's various microcomputer families with various optional modules. The

macro-assembler and text editor programs provided allow the designer to write and edit his programs in assembly language and then generate the machine language output necessary to program the 8748 EPROM. The availability of a high speed CRT and a diskette operating system eliminates the laborious input and output of paper tape files normally required during the assembly process. Finally, ICE 48 allows the user to extend the resources of his entire Intellec system into the 8048 socket of his own system and use all its emulation, debug, and display facilities directly.

### 1.3.7 Production

Once a working program has been achieved, a preproduction phase usually follows where several prototype systems are evaluated in simulated situations or in actual operation in the field. During this period the use of the 8748 EPROM allows quick alteration of the application program when problems or suggested changes arise. Depending on the magnitude and number of future changes anticipated, the first production units may also be shipped with EPROM processor. However, to achieve the maximum cost reduction potential in high volume applications, a conversion to the 8048 ROM is usually necessary. This is an easy transition since the 8048 and 8748 are pin and machine code compatible equivalents. The user merely develops a hexadecimal tape of his 8748 program memory contents using his Intellec System or PROMPT 48 development aid and sends it to Intel along with his 8048 order. As the 8048 ROM's arrive they can immediately replace the 8748 EPROMs.



## Chapter 2

# THE SINGLE COMPONENT MCS-48™ SYSTEM



## THE SINGLE COMPONENT MCS-48™ SYSTEM

2.0 Summary .....	2-1
2.1 Architecture .....	2-1
2.2 Pin Description .....	2-13
2.3 Programming, Verifying and Erasing EPROM .....	2-15
2.4 Test and Debug .....	2-17

# THE SINGLE COMPONENT MCS-48™ SYSTEM

## 2.0 Summary

The following sections describe in detail the functional characteristics of the 8748 EPROM, 8048 ROM and 8035 single component micro-computers. Unless otherwise noted, the following details apply to all three versions. This chapter is limited to those functions useful in single-chip implementations of the MCS-48. Chapter 3 discusses functions which allow expansion of program memory, data memory, and input-output capability.

### 2.1 Architecture

The following sections break the 8048 into functional blocks and describe each in detail.

#### 2.1.1 Arithmetic Section

The arithmetic section of the processor contains the basic data manipulation functions of the 8048 and can be divided into the following blocks:

- Arithmetic Logic Unit (ALU)
- Accumulator
- Carry Flag
- Instruction Decoder

In a typical operation data stored in the accumulator is combined in the ALU with data from another source on the internal bus (such as a register or I/O port) and the result is stored in the accumulator or another register. The following is a more detailed description of the function of each block:

#### Instruction Decoder

The operation code (op code) portion of each program instruction is stored in the Instruction Decoder and converted to outputs which control the function of each of the blocks of the Arithmetic Section. These lines control the source of data and the destination register as well as the function performed in the ALU.

#### Arithmetic Logic Unit

The ALU accepts 8-bit data words from one or two sources and generates an 8-bit result under control of the Instruction Decoder. The ALU can perform the following functions:

- Add With or Without Carry
- And, OR, Exclusive OR
- Increment/Decrement
- Bit Complement
- Rotate Left, Right
- Swap Nibbles
- BCD Decimal Adjust

If the operation performed by the ALU results in a value represented by more than 8 bits (overflow of most significant bit) a Carry Flag is set in the Program Status Word.

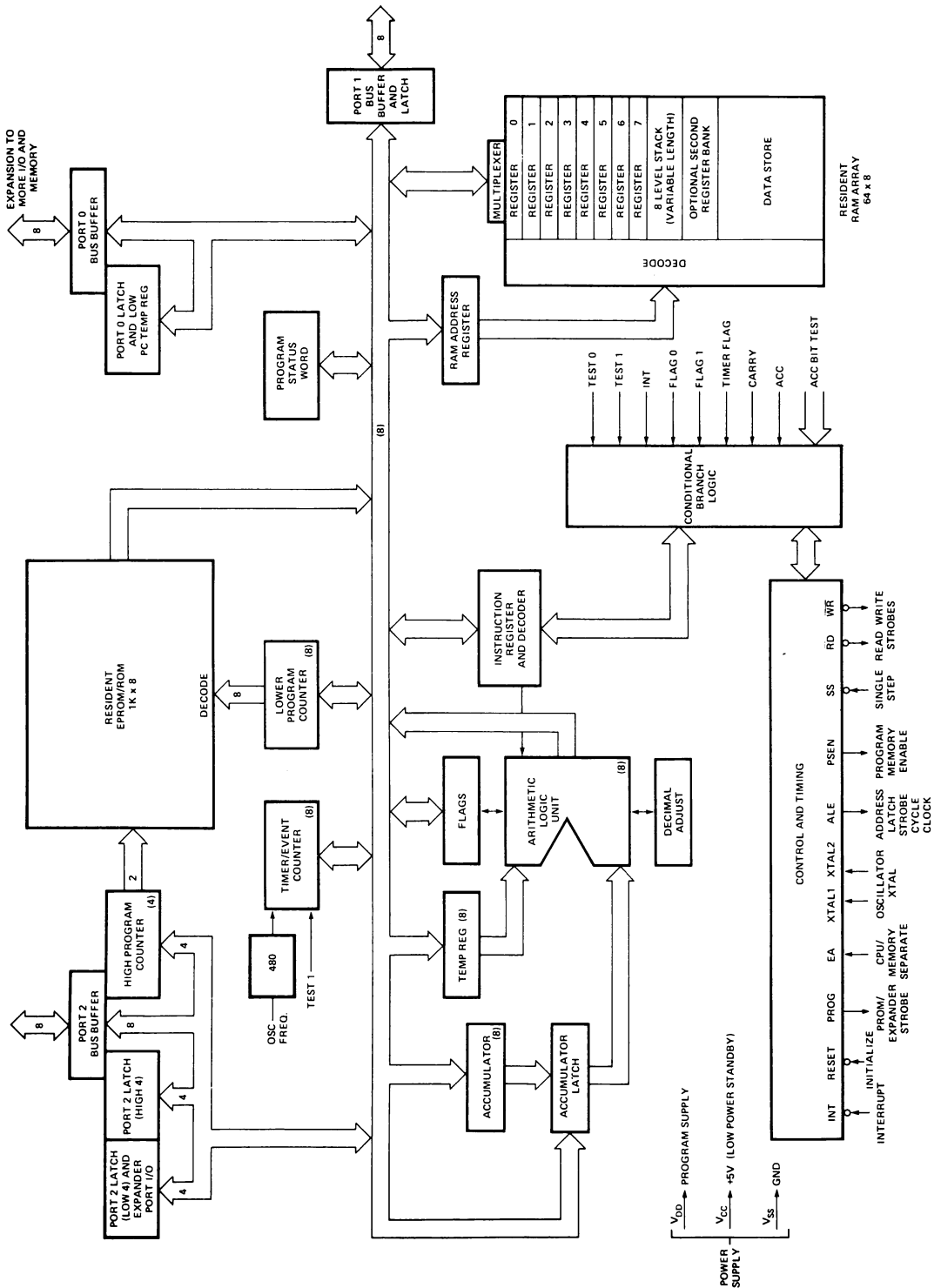
#### Accumulator

The accumulator is the single most important data register in the processor being one of the sources of input to the ALU and often the destination of the result of operations performed in the ALU. Data to and from I/O ports and memory also normally passes through the accumulator.

#### 2.1.2 Program Memory

Resident program memory consists of 1024 words eight bits wide which are addressed by the program counter. In the 8748 this memory is user programmable and erasable EPROM, in the 8048 the memory is ROM which is mask programmable at the factory, while the 8035 has no internal program memory and is used with external devices. Program code is completely interchangeable among the three versions. See Sec. 2.3 for EPROM programming techniques.

# SINGLE COMPONENT SYSTEM



8048 BLOCK DIAGRAM

There are three locations in Program Memory of special importance:

## LOCATION 0

Activating the Reset line of the processor causes the first instruction to be fetched from location 0.

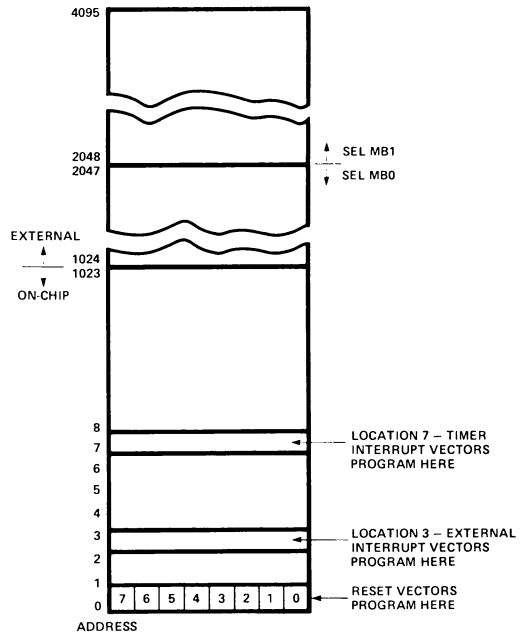
## LOCATION 3

Activating the Interrupt input line of the processor (if interrupt is enabled) causes a jump to subroutine.

## LOCATION 7

A timer/counter interrupt resulting from timer/counter overflow (if enabled) causes a jump to subroutine.

Therefore, the first instruction to be executed after initialization is stored in location 0, the first word of an external interrupt service subroutine is stored in location 3, and the first word of a timer/counter service routine is stored in location 7. Program memory can be used to store constants as well as program instructions. Instructions such as MOVP and MOV3 allow easy access to data "lookup" tables.

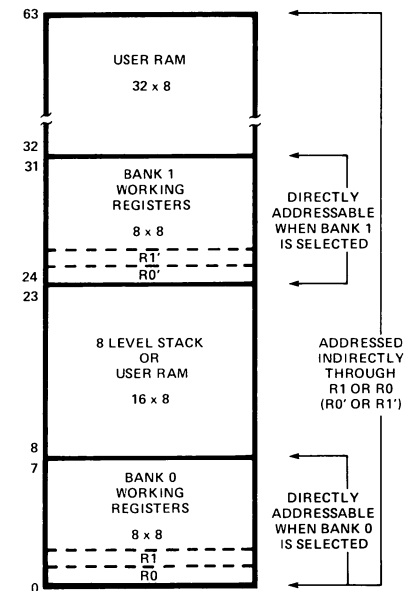


**MCS-48™ PROGRAM MEMORY MAP**

## 2.1.3 Data Memory

Resident data memory is organized as 64 words 8 bits wide. All 64 locations are indirectly addressable through either of two RAM Pointer Registers which reside at address 0 and 1 of the register array. In addition, the first 8 locations (0-7) of the array are designated as working registers and are directly addressable by several instructions. Since these registers are more easily addressed, they are usually used to store frequently accessed intermediate results. The DJNZ instruction makes very efficient use of the working registers as program loop counters by allowing the programmer to decrement and test the register in a single instruction.

By executing a Register Bank Switch instruction (SEL RB) RAM locations 24-31 are designated as the working registers in place of locations 0-7 and are then directly addressable. This second bank of working registers may be used as an extension of the first bank or reserved for use during interrupt service



**DATA MEMORY MAP**

subroutines allowing the registers of Bank 0 used in the main program to be instantly "saved" by a Bank Switch. Note that if this second bank is not used, locations 24-31 are still addressable as general purpose RAM. Since the two RAM pointer Registers R0 and R1 are a part of the working register array, bank switching effectively creates two more pointer registers (R0' and R1') which can be used with R0 and R1 to easily access up to four separate working areas in Ram at one time. RAM locations (8-23) also serve a dual role in that they contain the program counter stack as explained in Sec. 2.1.6. These locations are addressed by the Stack Pointer during subroutine calls as well as by RAM Pointer Registers R0 and R1. If the level of subroutine nesting is less than 8, all stack registers are not required and can be used as general purpose RAM locations. Each level of subroutine nesting not used provides the user with two additional RAM locations.

## 2.1.4 Input/Output

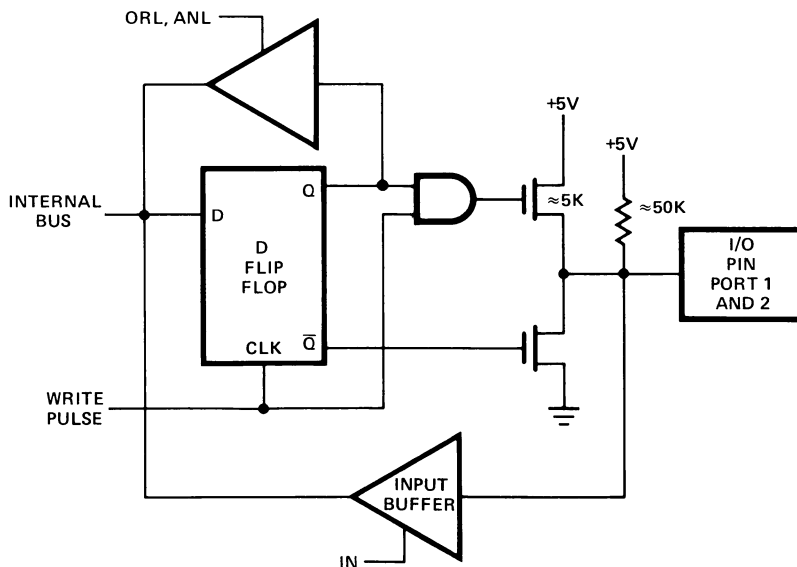
The 8048 has 27 lines which can be used for input or output functions. These lines are grouped as 3 ports of 8 lines each which serve as either inputs, outputs or bidirectional

ports and 3 "test" inputs which can alter program sequences when tested by conditional jump instructions.

## Ports 1 and 2

Ports 1 and 2 are each 8 bits wide and have identical characteristics. Data written to these ports is statically latched and remains unchanged until rewritten. As input ports these lines are non latching, i.e., inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

The lines of ports 1 and 2 are called quasi-bidirectional because of a special output circuit structure which allows each line to serve as an input, an output, or both even though outputs are statically latched. The figure shows the circuit configuration in detail. Each line is continuously pulled up to +5v through a resistive device of relatively high impedance ( $\sim 50K\Omega$ ). This pullup is sufficient to provide the source current for a TTL high level yet can be pulled low by a standard TTL gate thus allowing the same pin to be used for both input and output. To provide fast switching times in a "0" to "1" transition a relatively low



**"QUASI BI DIRECTIONAL" PORT STRUCTURE**



impedance device ( $\sim 5K\Omega$ ) is switched in momentarily ( $\sim 500ns$ ) whenever a "1" is written to the line. When a "0" is written to the line a low impedance ( $\sim 3K\Omega$ ) device overcomes the light pullup and provides TTL current sinking capability. Since the pulldown transistor is a low impedance device a "1" must first be written to any line which is to be used as an input. Reset initializes all lines to the high impedance "1" state. This structure allows input and output on the same pin and also allows a mix of input lines and output lines on the same port. The quasi-bidirectional port in combination with the ANL and ORL logical instructions provide an efficient means for handling single line inputs and outputs within an 8-bit processor.

### Bus

Bus is also an 8-bit port which is a true bidirectional port with associated input and output strobes. If the bidirectional feature is not needed, Bus can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed however.

As a static port, data is written and latched using the OUTL instruction and inputted using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding RD and WR output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port the MOVX instructions are used to read and write the port. A write to the port generates a pulse on the WR output line and output data is valid at the trailing edge of WR. A read of the port generates a pulse on the RD output line and input data must be valid at the trailing edge of RD. When not being written or read, the BUS lines are in a high impedance state.

### 2.1.5 Test and INT Inputs

Three pins serve as inputs and are testable with the conditional jump instruction. These are T0, T1, and  $\overline{INT}$ . These pins allow inputs

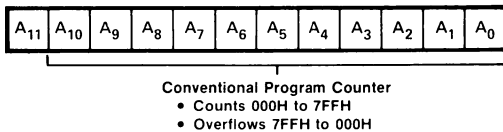
to cause program branches without the necessity to load an input port into the accumulator. The T0, T1, and  $\overline{INT}$  pins have other possible functions as well. See the pin description in Sec. 2.2.

### 2.1.6 Program Counter and Stack

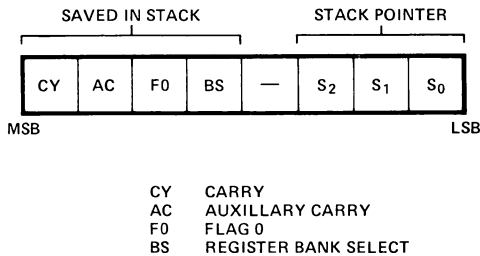
The Program Counter is an independent counter while the Program Counter Stack is implemented using pairs of registers in the Data Memory Array. Only 10 bits of the Program Counter are used to address the 1024 words of on-board program memory while the most significant two bits are used for external Program Memory fetches. The Program Counter is initialized to zero by activating the Reset line.

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the Program Counter Stack. The pair to be used is determined by a 3-bit Stack Pointer which is part of the Program Status Word (PSW). Data RAM locations 8 thru 23 are available as stack registers and are used to store the Program Counter and 4 bits of PSW as shown in the figure. The Stack Pointer when initialized to 000 points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL. Nesting of subroutines within subroutines can continue up to 8 times without overflowing the stack. If overflow does occur the deepest address stored (location 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

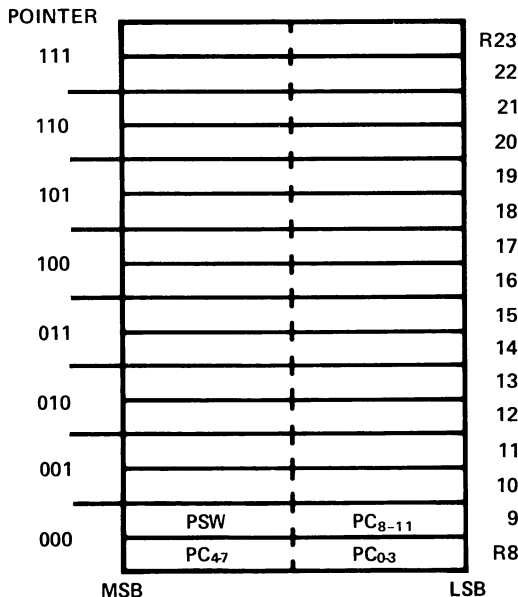
The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the Stack Pointer to be decremented and the contents of the resulting register pair to be transferred to the Program Counter.



## PROGRAM COUNTER



## PROGRAM STATUS WORD (PSW)



## PROGRAM COUNTER STACK

### 2.1.7 Program Status Word

An 8-bit status word which can be loaded to and from the accumulator exists called the Program Status Word (PSW). The accompanying figure shows the information available in the word. The Program Status Word is actually a collection of flip-flops throughout the machine which can be read or written as a whole. The ability to write to PSW allows for easy restoration of machine status after a power down sequence.

The upper four bits of PSW are stored in the Program Counter Stack with every jump to subroutine or interrupt vector and are optionally restored upon return with the RETR instruction. The RET return instruction does not update PSW.

The PSW bit definitions are as follows:

- Bits 0 - 2: Stack Pointer bits ( $S_0$ ,  $S_1$ ,  $S_2$ )
- Bit 3: Not used ("1" level when read)
- Bit 4: Working Register Bank Switch Bit (BS)  
0 = Bank 0  
1 = Bank 1
- Bit 5: Flag 0 bit (F0) user controlled flag which can be complemented or cleared, and tested with the conditional jump instruction JF0.
- Bit 6: Auxiliary Carry (AC) carry bit generated by an ADD instruction and used by the decimal adjust instruction DA A.
- Bit 7: Carry (CY) carry flag which indicates that the previous operation has resulted in overflow of the accumulator.

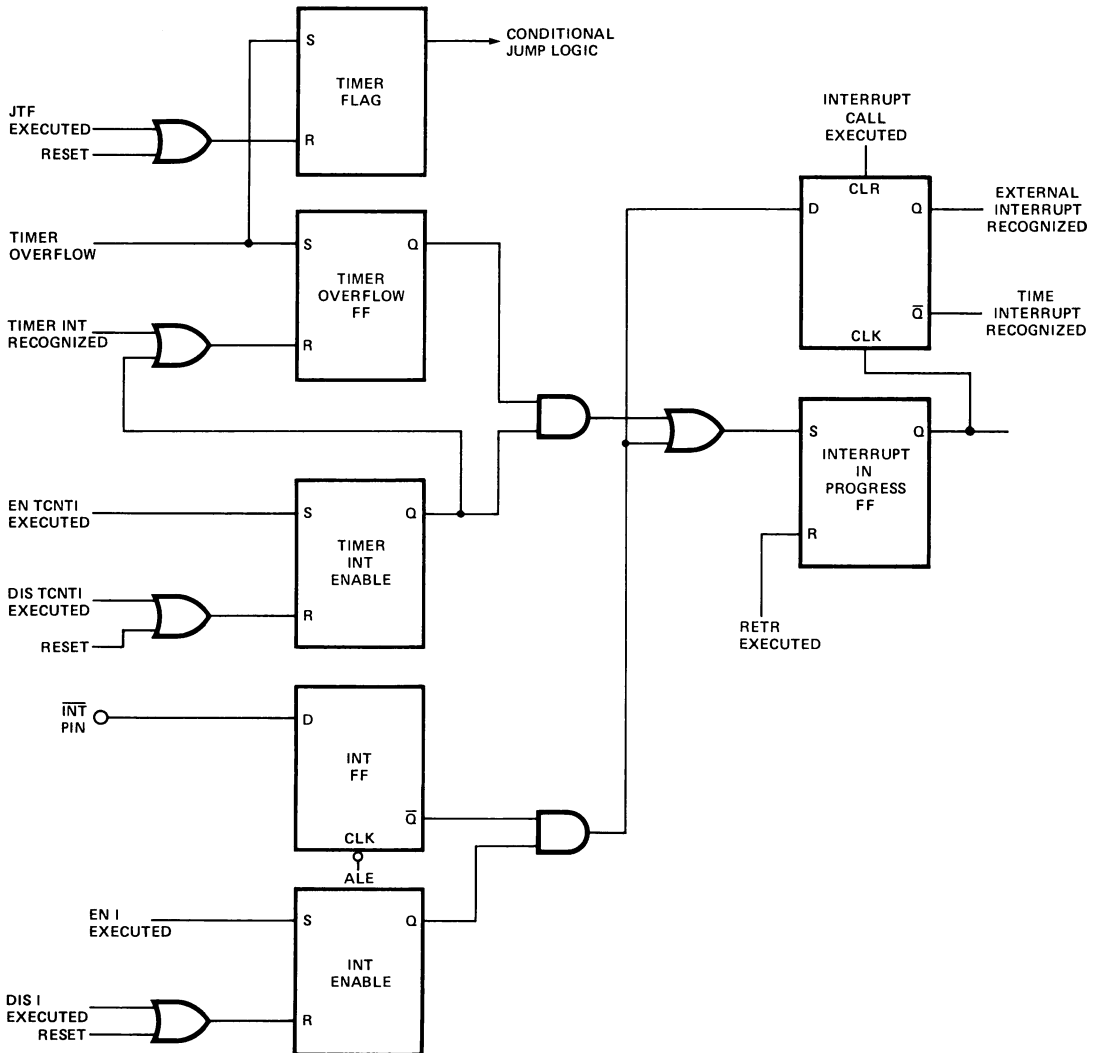
### 2.1.8 Conditional Branch Logic

The conditional branch logic within the processor enables several conditions internal and external to the processor to be tested by the users program. By using the conditional jump instruction the following conditions can effect a change in the sequence of the program execution.

Device Testable	Jump Conditions (Jump On)	
	All zeros	not all zeros
Accumulator	—	1
Accumulator Bit	—	1
Carry Flag	0	1
User Flags (F0, F1)	—	1
Timer Overflow Flag	—	1
Test Inputs (T0, T1)	0	1
Interrupt Input (INT)	0	—

### 2.1.9 Interrupt

An interrupt sequence is initiated by applying a low "0" level input to the INT pin. Interrupt is level triggered and active low to allow "WIRE ORing" of several interrupt sources at the input pin. The Interrupt line is sampled every machine cycle during ALE and when detected causes a "jump to subroutine" at location 3 in program memory as soon as all cycles of the current instruction are complete. As in any CALL to subroutine, the Program Counter



and Program Status word are saved in the stack. For a description of this operation see the previous section, Program Counter and Stack. Program Memory location 3 usually contains an unconditional jump to an interrupt service subroutine elsewhere in program memory. The end of an interrupt service subroutine is signalled by the execution of a Return and Restore Status instruction RETR. The interrupt system is single level in that once an interrupt is detected all further interrupt requests are ignored until execution of an RETR re-enables the interrupt input logic. This occurs at the beginning of the second cycle of the RETR instruction. This sequence holds true also for an internal interrupt generated by timer overflow. If an internal timer/counter generated interrupt and an external interrupt are detected at the same time, the external source will be recognized. See the following Timer/Counter section for a description of timer interrupt. If needed, a second external interrupt can be created by enabling the timer/counter interrupt, loading FFH in the Counter (one less than terminal count), and enabling the event counter mode. A "1" to "0" transition on the T1 input will then cause an interrupt vector to location 7.

### Interrupt Timing

The interrupt input may be enabled or disabled under Program Control using the EN I and DIS I instructions. Interrupts are disabled by Reset and remain so until enabled by the users program. An interrupt request must be removed before the RETR instruction is executed upon return from the service routine otherwise the processor will re-enter the service routine immediately. Many peripheral devices prevent this situation by resetting their interrupt request line whenever the processor accesses (Reads or Writes) the peripherals data buffer register. If the interrupting device does not require access by the processor, one output line of the 8048 may be designated as an "interrupt acknowledge" which is activated by the service subroutine to reset the interrupt request. The INT pin may also be tested using the conditional jump instruction JNI. This instruction may be used

to detect the presence of a pending interrupt before interrupts are enabled. If interrupt is left disabled, INT may be used as another test input like T0 and T1.

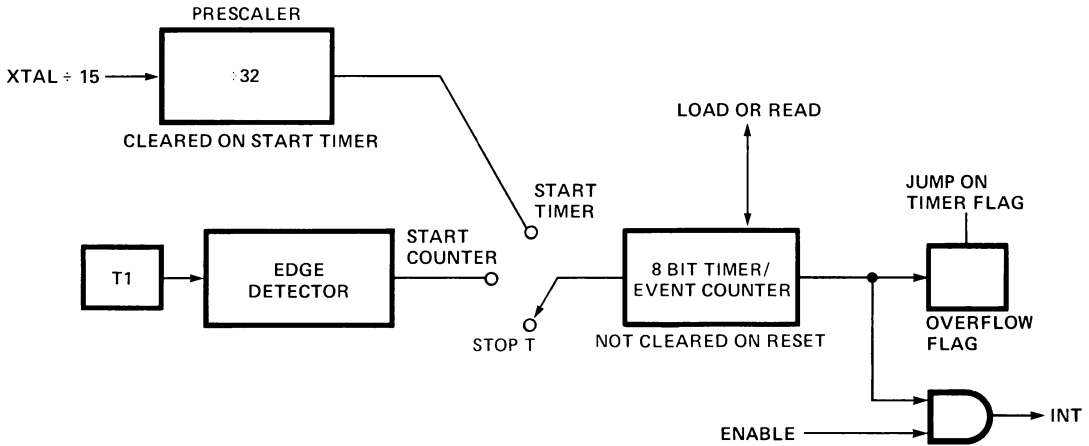
### 2.1.10 Timer/Counter

The 8048 contains a counter to aid the user in counting external events and generating accurate time delays without placing a burden on the processor for these functions. In both modes the counter operation is the same, the only difference being the source of the input to the counter.

#### Counter

The 8-bit up binary counter is presettable and readable with two MOV instructions which transfer the contents of the accumulator to the counter and vice versa. The counter content is not affected by Reset and is initialized solely by the MOV T,A instruction. The counter is stopped by a Reset or STOP TCNT instruction and remains stopped until started as a timer by a START T instruction or as an event counter by a START CNT instruction. Once started the counter will increment to its maximum count (FF) and overflow to zero continuing its count until stopped by a STOP TCNT instruction or Reset.

The increment from maximum count to zero (overflow) results in the setting of an overflow flag flip-flop and in the generation of an interrupt request. The state of the overflow flag is testable with the conditional jump instruction JTF. The flag is reset by executing a JTF or by Reset. The interrupt request is stored in a latch and then ORed with the external interrupt input INT. The timer interrupt may be enabled or disabled independently of external interrupt by the EN TCNTI and DIS TCNTI instructions. If enabled, the counter overflow will cause a subroutine call to location 7 where the timer or counter service routine may be stored. If timer and external interrupts occur simultaneously, the external source will be recognized and the Call will be to location 3. Since the timer interrupt is latched it will remain pending until the external device is serviced and immediately be recognized upon return for the service routine. The pending



## TIMER/EVENT COUNTER

timer interrupt is reset by the Call to location 7 or may be removed by executing a DIS TCNTI instruction.

### As an Event Counter

Execution of a START CNT instruction connects the T1 input pin to the counter input and enables the counter. Subsequent high to low transitions on T1 will cause the counter to increment. The maximum rate at which the counter may be incremented is once per three instruction cycles (every  $7.5\mu\text{sec}$  when using a 6MHz crystal)—there is no minimum frequency. T1 input must remain high for at least 100ns after each transition.

### As a Timer

Execution of a START T instruction connects an internal clock to the counter input and enables the counter. The internal clock is derived by passing the basic 400 KHz machine cycle clock ALE through a  $\div 32$  prescaler. The prescaler is reset during the START T instruction. The resulting 12.5 KHz clock increments the counter every  $80\mu\text{sec}$  (assuming 6 MHz XTAL). Various delays between  $80\mu\text{sec}$  and 20 msec (256 counts) can be obtained by presetting the counter and detecting overflow. Times longer than 20 msec may be achieved by accumulating mul-

iple overflows in a register under software control. For time resolution less than  $80\mu\text{sec}$  an external clock can be applied to the T1 input and the counter operated in the event counter mode. ALE divided by 3 or more can serve as this external clock. Very small delays or “fine tuning” of larger delays can be easily accomplished by software delay loops.

### 2.1.11 Clock and Timing Circuits

Timing generation for the 8048 is completely self-contained with the exception of a frequency reference which can be XTAL, series RC, or external clock source. The Clock and Timing circuitry can be divided into the following functional blocks:

#### Oscillator

The on-board oscillator is a high gain series resonant circuit with a frequency range of 1 to 6MHz. The X1 external pin is the input to the amplifier stage while X2 is the output. A crystal or series RC connected between X1 and X2 provides the feedback and phase shift required for oscillation. A 5.994 MHz crystal provides for easy deviation of all standard communications frequencies. If an accurate frequency reference and maximum processor speed are not required, a resistor and

capacitor may be used in place of the crystal. With an RC the oscillator frequency will be approximately 3 MHz. For higher speed operation a crystal should be used. An externally generated clock may also be applied to X1 as the frequency source.

## State Counter

The output of the oscillator is divided by 3 in the State Counter to create a clock which defines the state times of the machine (CLK). CLK can be made available on the external pin T0 by executing an ENTO CLK instruction. The output of CLK on T0 is disabled by Reset of the processor.

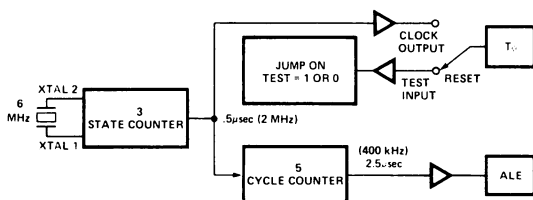
## Cycle Counter

CLK is then divided by 5 in the Cycle Counter to provide a clock which defines a machine cycle consisting of 5 machine states. This clock is called Address Latch Enable (ALE) because of its function in MCS-48 systems with external memory. It is provided continuously on the ALE output pin.

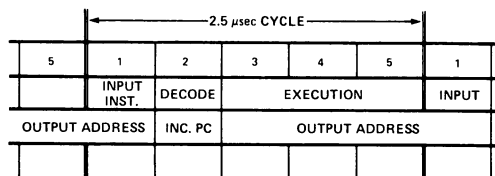
### 2.1.12 Reset

The reset input provides a means for initialization for the processor. This Schmitt-trigger input has an internal pullup resistor which in combination with an external 1  $\mu$ f capacitor provides an internal reset pulse of sufficient length to guarantee all circuitry is reset. If the

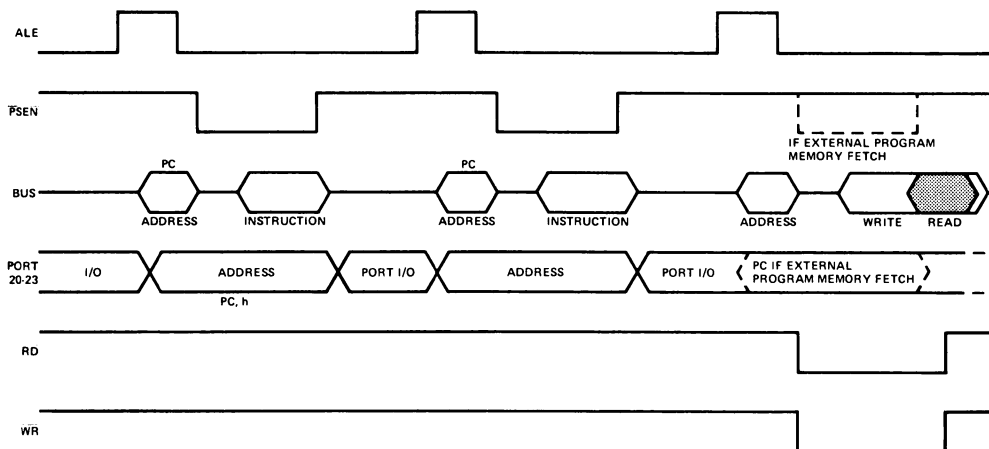
## DIAGRAM OF 8048 CLOCK UTILITIES



## INSTRUCTION CYCLE

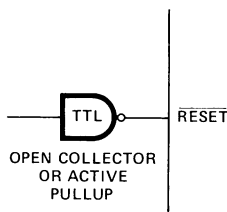


## MCS-48™ CYCLE TIMING FOR EXTERNAL MEMORY

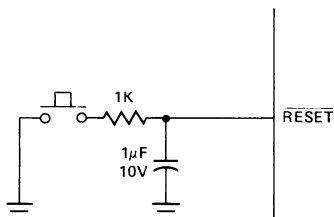


reset pulse is generated externally the reset pin must be held at ground (.5V) for at least 50 milliseconds after the power supply is within tolerance.

## EXTERNAL RESET



## POWER ON RESET



Reset performs the following functions:

1. Sets program counter to zero.
2. Sets stack pointer to zero.
3. Selects register bank 0.
4. Selects memory bank 0.
5. Sets BUS to high impedance state.
6. Sets Ports 1 and 2 to input mode.
7. Disables interrupts (timer and external)
8. Stops timer.
9. Clears timer flag.
10. Clears F0 and F1.
11. Disables clock output from T0.

### 2.1.13 Single-Step

This feature provides the user with a debug capability in that the processor can be stepped through the program one instruction at a time. While stopped, the address of the next instruction to be fetched is available concurrently on BUS and the lower half of Port 2. The user can therefore follow the program through each of the instruction steps. A timing diagram, showing the interaction between output ALE and input SS is shown. The BUS buffer contents are lost during single step, however, a latch may be added to re-establish the lost I/O capability if needed. (See 2.4.1).

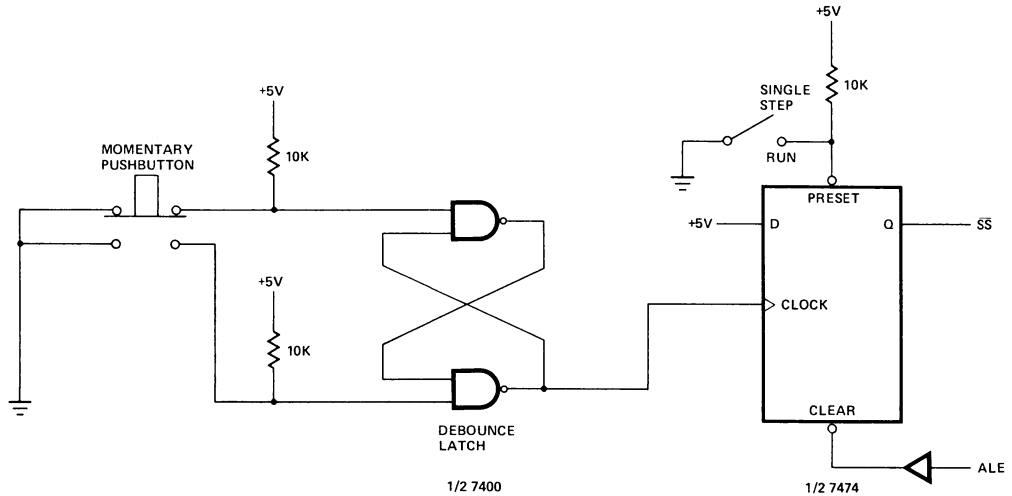
## Timing

The 8048 operates in a single-step mode as follows:

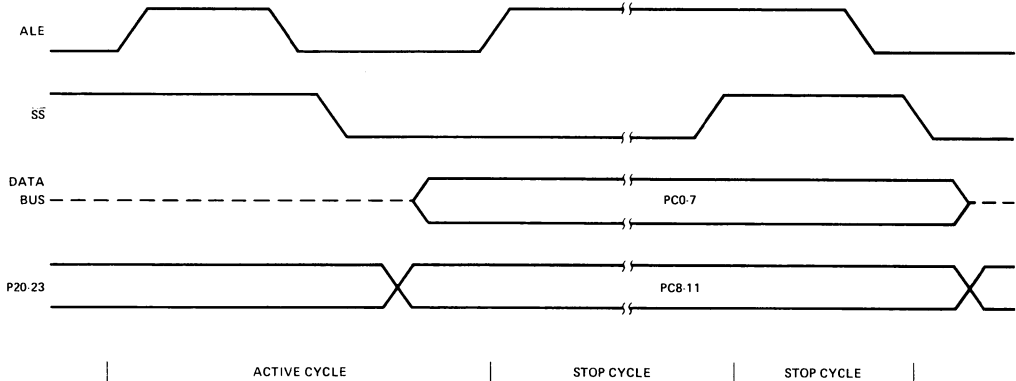
1. The processor is requested to stop by applying a low level on SS.
2. The processor responds by stopping during the instruction fetch portion of the next instruction. If a double cycle instruction is in progress when the single step command is received, both cycles will be completed before stopping.
3. The processor acknowledges it has entered the stopped state by raising ALE high. In this state (which can be maintained indefinitely) the address of the next instruction to be fetched is present on BUS and the lower half of port 2.
4. SS is then raised high to bring the processor out of the stopped mode allowing it to fetch the next instruction. The exit from stop is indicated by the processor bringing ALE low.
5. To stop the processor at the next instruction SS must be brought low again as soon as ALE goes low. If SS is left high the processor remains in a "Run" mode.

A diagram for implementing the single step function of the 8748 is shown. A D-type flip-flop with preset and clear is used to generate SS. In the run mode SS is held high by keeping the flip-flop preset (preset has precedence over the clear input). To enter single step, preset is removed allowing ALE to bring SS low via the clear input. ALE should be buffered since the clear input of an SN7474 is the equivalent of 3 TTL loads. The processor is now in the stopped state. The next instruction is initiated by clocking a "1" into the flip-flop. This "1" will not appear on SS unless ALE is high removing clear from the flip-flop. In response to SS going high the processor begins an instruction fetch which brings ALE low resetting SS through the clear input and causing the processor to again enter the stopped state.

## SINGLE STEP CIRCUIT



## SINGLE STEP TIMING

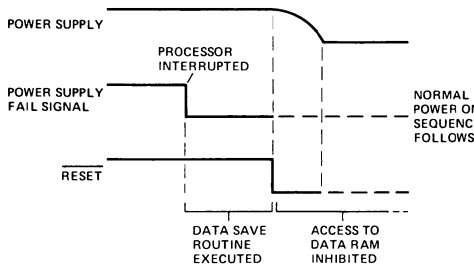


### 2.1.14 Power Down Mode (8048 ROM version only)

Extra circuitry has been added to the 8048 ROM version to allow power to be removed from all but the 64 x 8 data ram array for low power standby operation. In the power down mode the contents of data ram can be maintained while drawing typically 10 to 15% of normal operating power requirements.

$V_{CC}$  serves as the 5V supply pin for the bulk of 8048 circuitry while the  $V_{DD}$  pin supplies only the RAM array. In normal operation both pins are at 5V while in standby  $V_{CC}$  is at ground and only  $V_{DD}$  is maintained at 5V. Applying Reset to the processor through the Reset pin inhibits any access to the RAM by the processor and guarantees that RAM cannot be inadvertently altered as power is removed from  $V_{CC}$ .





## POWER DOWN SEQUENCE

A typical power down sequence occurs as follows:

1. Imminent power supply failure is detected by user defined circuitry. Signal must be early enough to allow 8048 to save all necessary data before  $V_{CC}$  falls below normal operating limits.
2. Power fail signal is used to interrupt processor and vector it to a power fail service routine.
3. Power fail routine saves all important data and machine status in the internal data RAM array. Routine may also initiate transfer of backup supply to the  $V_{DD}$  pin and indicate to external circuitry that power fail routine is complete.
4. Reset is applied to guarantee data will not be altered as the power supply falls out of limits. Reset must be held low until  $V_{CC}$  is at ground level.

Recovery from the Power Down mode can occur as any other power-on sequence with an external capacitor on the Reset input providing the necessary delay. See the previous section on Reset.

### 2.1.15 External Access Mode

Normally the first 1K words of program memory are automatically fetched from internal ROM or EPROM. The EA input pin however allows the user to effectively disable internal

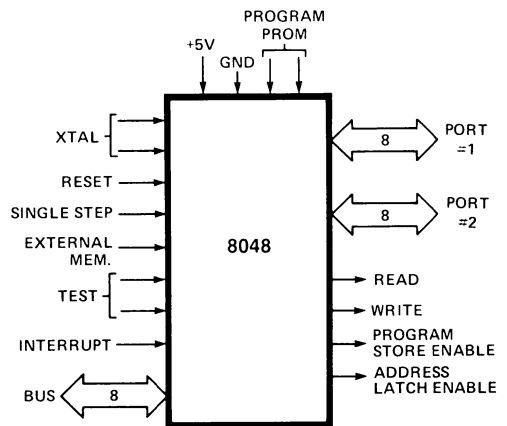
program memory by forcing all program memory fetches to reference external memory. The following chapter explains how access to external program memory is accomplished.

The External Access mode is very useful in system test and debug because it allows the user to disable his internal applications program and substitute an external program of his choice—a diagnostic routine for instance. In addition, the section on Test and Debug explains how internal program memory can be read externally, independent of the processor.

A "1" level on EA initiates the external access mode. For proper operation, Reset should be applied while the EA input is changed.

## 2.2 Pin Description

The 8048 and 8748 are packaged in 40 pin Dual In-Line Packages (DIP's). The following is a summary of the functions of each pin. Where it exists, the second paragraph describes each pin's function in an expanded MCS-48 system. Unless otherwise specified, each input is TTL compatible and each output will drive one standard TTL load.



## 8048 LOGIC SYMBOL

## SINGLE COMPONENT SYSTEM

Designation	Pin Number	Function
V <sub>SS</sub>	20	Circuit GND potential
V <sub>DD</sub>	26	Programming power supply; +25V during program, +5V during operation for both ROM and PROM. Low power standby pin in 8048 ROM version
V <sub>CC</sub>	40	Main power supply; +5V during operation and 8748 programming.
PROG	25	Program pulse (+25V) input pin during 8748 programming. Output strobe for 8243 I/O expander.
P10-P17 (Port 1)	27-34	8-bit quasi-bidirectional port.
P20-P27 (Port 2)	21-24 35-38	8-bit quasi-bidirectional port.  P20-P23 contain the four high order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243.
D0-D7 (BUS)	12-19	True bidirectional port which can be written or read synchronously using the $\overline{RD}$ , $\overline{WR}$ strobes. The port can also be statically latched.  Contains the 8 low order program counter bits during an external program memory fetch, and receives the addressed instruction under the control of PSEN. Also contains the address and data during an external RAM data store instruction, under control of ALE, $\overline{RD}$ , and $\overline{WR}$ .
T0	1	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 can be designated as a clock output using ENTO CLK instruction. T0 is also used during programming.
T1	39	Input pin testable using the JT1, and JNT1 instructions. Can be designated the event counter input using the STRT CNT instruction.
$\overline{INT}$	6	Interrupt input. Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. (Active low)
$\overline{RD}$	10	Output strobe activated during a BUS read. Can be used to enable data onto the BUS from an external device. (Active low)  Used as a Read Strobe to External Data Memory.

Designation	Pin Number	Function
$\overline{\text{RESET}}$	4	Input which is used to initialize the processor. Also used during PROM programming and verification. (Active low)
$\overline{\text{WR}}$	8	Output strobe during a BUS write. (Active low) Used as write strobe to external data memory.
ALE	11	Address Latch Enable. This signal occurs once during each cycle and is useful as a clock output. The negative edge of ALE strobes address into external data and program memory.
$\overline{\text{PSEN}}$	9	Program Store Enable. This output occurs only during a fetch to external program memory. (Active Low)
$\overline{\text{SS}}$	5	Single step input can be used in conjunction with ALE to "single step" the processor through each instruction. (Active Low)
EA	7	External Access input which forces all program memory fetches to reference external memory. Useful for emulation and debug, and essential for testing and program verification. (Active High)
XTAL1	2	One side of crystal input for internal oscillator. Also input for external source.
XTAL2	3	Other side of crystal input.

### 2.3 Programming, Verifying and Erasing EPROM

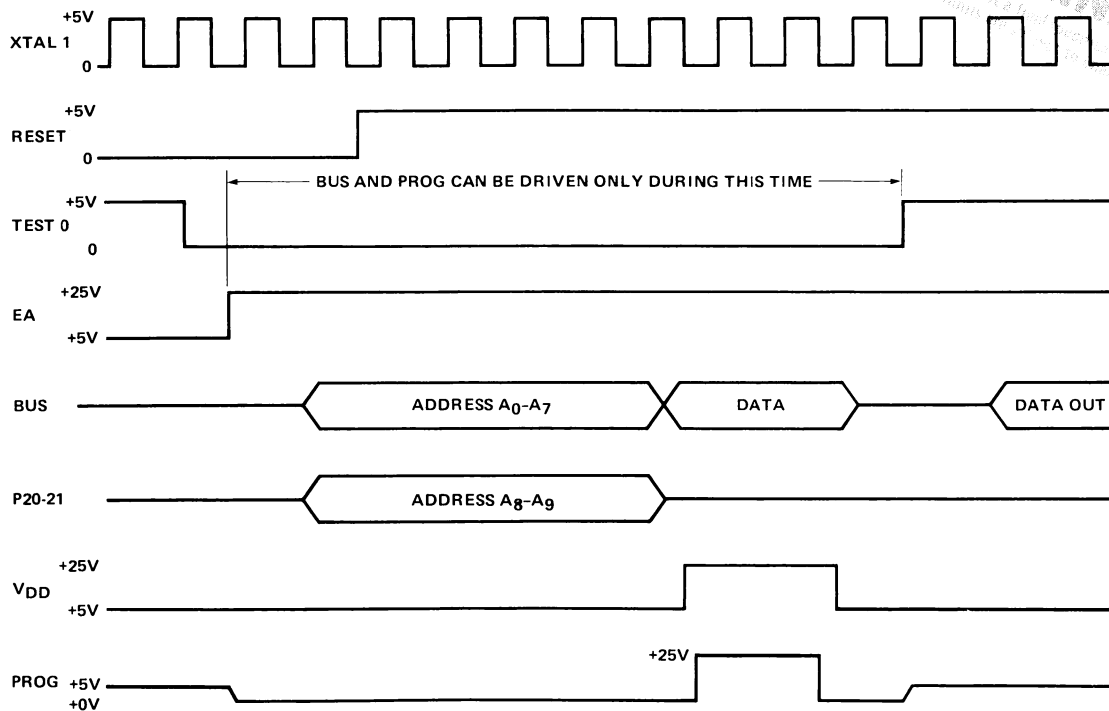
The internal Program Memory of the 8748 may be erased and reprogrammed by the user as explained in the following sections:

#### 2.3.1 Programming/Verification

In brief, the programming process consists of: activating the program mode, applying an address, latching the address, applying data, and applying a programming pulse. Each word is programmed completely before moving on to the next and is followed by a verification step. The following is a list of the pins used for programming and a description of their functions:

Pin	Function
XTAL 1	Clock Input (1 to 6MHz)
Reset	Initialization and Address Latching
Test 0	Selection of Program or Verify Mode
EA	Activation of Program/Verify Modes
BUS	Address and Data Input Data Output During Verify
P20-1	Address Input
V <sub>DD</sub>	Programming Power Supply
PROG	Program Pulse Input

## SINGLE COMPONENT SYSTEM



**WARNING:** An attempt to program a missocketed 8748 will result in severe damage to the part. An indication of a properly socketed part is the appearance of the ALE clock output. The lack of this clock may be used to disable the programmer.

## PROGRAMMING/VERIFY SEQUENCE

The detailed Program/Verify sequence is:

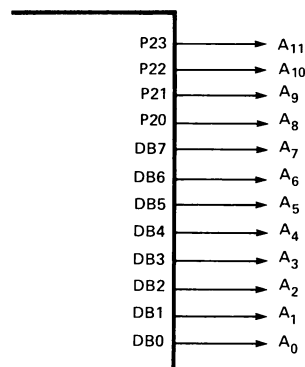
1.  $V_{DD} = 5v$ , Clock applied, Reset = 0v  
Test 0 = 5v, EA = 5v, BUS and PROG floating
2. Insert 8748 in programming socket
3. Test 0 = 0v (Select Program Mode)
4. EA = 25v (Activate Program Mode)
5. Address applied to BUS and P20-1
6. Reset = 5v (Latch Address)
7. Data applied to BUS
8.  $V_{DD} = 25v$  (Programming Power)
9. PROG = 0v followed by one 50ms pulse to 25v
10.  $V_{DD} = 5v$
11. TEST 0 = 5v (Verify Mode)
12. Read and Verify Data on BUS
13. TEST 0 = 0v
14. Reset = 0v and repeat from Step 5
15. Programmer should be at conditions of Step 1 when 8748 is removed from socket.

## 2.4 Test and Debug

Several MCS-48 features described in the previous sections are discussed here to emphasize their use in testing MCS-48 components and in debugging MCS-48 based systems.

### 2.4.1 Single Step

Single step circuitry within the micro-computer in combination with the external circuitry described in Section 2.1.13 allows the user to execute one instruction at a time whether the instruction is one or two cycles in length. After completion of the instruction the processor halts with the address of the next instruction to be fetched available on the eight lines of BUS and the lower 4-bits of port 2.



### ADDRESS OUTPUT DURING SINGLE STEP

This allows the user to step through his program and note the sequence of instructions being executed.

While the processor is stopped, the I/O information on BUS and the 4-bits of port 2 is, of course, not available. I/O information is, however, valid at the leading edge of ALE and can be latched externally using this signal if necessary.

### 2.4.2 Disabling Internal Program Memory

Applying +5V to the EA (external access) pin of the MCS-48 microcomputers allows the user to effectively disable internal program memory by forcing all instruction fetches to occur from an external memory. This external memory can be connected as explained in the section on program memory expansion and can contain a diagnostic routine to exercise the processor, the internal RAM, the timer, and the I/O lines. EA should be switched only when the processor is in RESET.

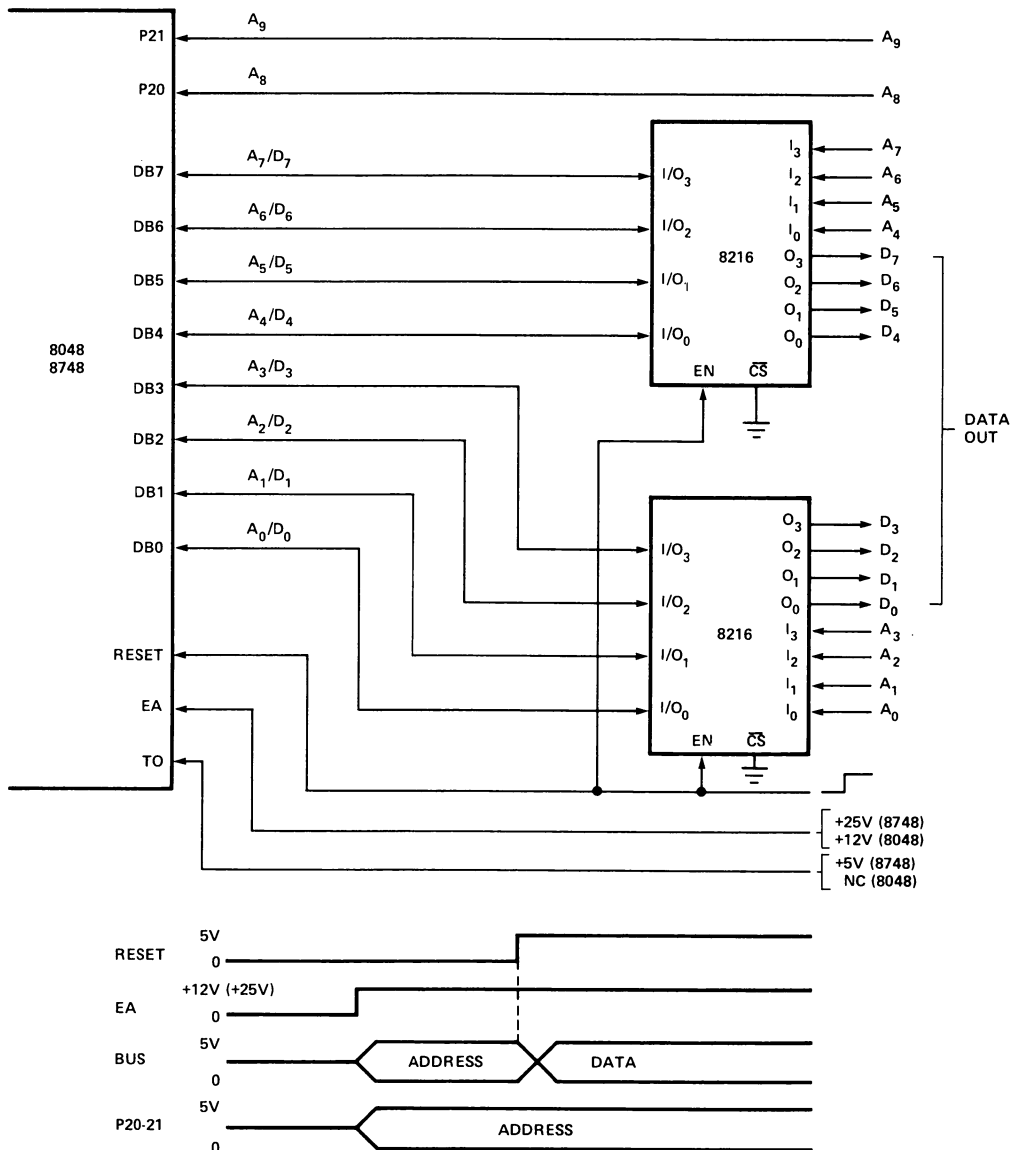
### 2.4.3 Reading Internal Program Memory

Just as the processor may be isolated from internal program memory using EA, program memory can be read independent of the processor using the verification mode described in the previous section, Programming/Verification.

## SINGLE COMPONENT SYSTEM

The processor is placed in the READ mode by applying a high voltage (+25V for the 8748, +12V for the 8048) to the EA pin and +5V to the T0 (8748 only) input pin. RESET must be at 0V when voltage is applied to EA. The address of the location to be read is then applied to the same lines (TTL levels) of BUS

and Port 2 which output the address during single step (see below). The address is latched by a "0" to "1" transition on RESET and a high level on RESET causes the contents of the program memory location addressed to appear on the eight lines of BUS.



## READING INTERNAL PROGRAM MEMORY

## Chapter 3

# THE EXPANDED MCS-48™ SYSTEM



## THE EXPANDED MCS-48™ SYSTEM

3.0 Summary .....	3-1
3.1 Expansion of Program Memory .....	3-1
3.2 Expansion of Data Memory .....	3-4
3.3 Expansion of Input/Output .....	3-5
3.4 Multi-Chip MCS-48 Systems .....	3-9
3.5 Memory Bank Switching .....	3-10



# THE EXPANDED MCS-48™ SYSTEM

## 3.0 Summary

If the capabilities resident on the single-chip 8048, 8748, or 8035 are not sufficient for your system requirements, special on-board circuitry allows the addition of a wide variety external memory, I/O, or special peripherals you may require. The processors can be directly and simply expanded in the following areas:

- Program Memory to 4K words
- Data Memory to 320 words
- I/O by unlimited amount
- Special Functions using 8080 peripherals

By using bank switching techniques maximum capability is essentially unlimited. Bank switching is discussed later in the chapter. Expansion is accomplished in two ways:

1. Expander I/O—A special I/O Expander circuit the 8243 provides for the addition of four 4-bit Input/Output ports with the sacrifice of only the lower half (4 bits) of port 2 for inter-device communication. Multiple 8243's may be added to this 4-bit bus by generating the required "chip select" lines.
2. Standard 8080 Bus—One port of the 8048 is like the 8 bit bidirectional data bus of the 8080A microcomputer system allowing interface to the numerous standard memories and peripherals of the MCS-80 microcomputer family.

MCS-48 systems can be configured using either or both of these expansion features to optimize system capabilities to the application. Both expander devices and standard memories and peripherals can be added in virtually any number and combination required.

## 3.1 Expansion of Program Memory

Program Memory is expanded beyond the resident 1K words by using the 8080 BUS feature of the MCS-48. All program memory fetches from addresses less than 1024 occur internally with no external signals being generated (except ALE which is always present). At address 1024 the 8048 automatically initiates external program memory fetches.

### 3.1.1 Instruction Fetch Cycle (External)

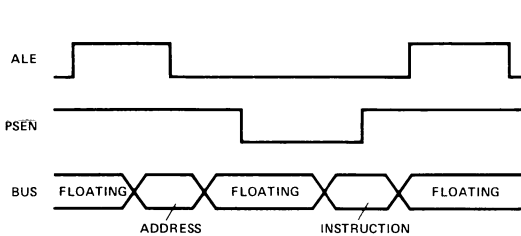
For all instruction fetches from addresses of 1024 or greater the following will occur:

1. The contents of the 12 bit program counter will be output on BUS and the lower half of port 2.
2. Address Latch Enable (ALE) will indicate the time at which address is valid. The trailing edge of ALE is used to latch the address externally.
3. Program Store Enable ( $\overline{\text{PSEN}}$ ) indicates that an external instruction fetch is in progress and serves to enable the external memory device.
4. BUS reverts to input mode and the processor accepts its 8 bit contents as an instruction word.

All instruction fetches including those of addresses less than 1024 can be forced to be external by activating the EA pin of the 8048. The 8035 processor without program memory always operates in the external program memory mode (EA=5V).

### 3.1.2 Extended Program Memory Addressing (Beyond 2K)

For programs of 2K words or less, the 8048 addresses program memory in the conventional manner. Addresses beyond 2047 can be reached by executing a program memory



### INSTRUCTION FETCH FROM EXTERNAL PROGRAM MEMORY

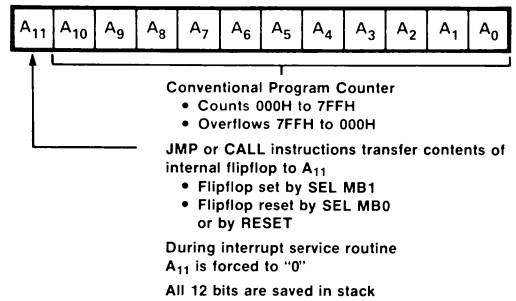
bank switch instruction (SEL MB0, SEL MB1) followed by a branch instruction (JMP or CALL). The bank switch feature extends the range of branch instructions beyond their normal 2K range and at the same time prevents the user from inadvertently crossing the 2K boundary.

#### Program Memory Bank Switch

The switching of 2K program memory banks is accomplished by directly setting or resetting the most significant bit of the program counter (bit 11). Bit 11 is not altered by normal incrementing of the program counter but is loaded with the contents of a special flip-flop each time a branch instruction is executed. This special flip-flop is set by executing an SEL MB1 instruction and reset by SEL MB0. Therefore, the SEL MB instruction may be executed at any time prior to the actual bank switch which occurs during the next branch instruction encountered. Since all twelve bits of the program counter including bit (11) are stored in the stack when a Call is executed, the user may jump to subroutines across the 2K boundary and the proper bank will be restored upon return. However, the bank switch flipflop will not be altered on return.

#### Interrupt Routines

Interrupts always vector the program counter to location 3 or 7 in the first 2K bank and bit 11 of the program counter is held at "0" during the interrupt service routine. The end of the service routine is signalled by the execution of an RETR instruction. Interrupt service routines should therefore be contained



### PROGRAM COUNTER

entirely in the lower 2K words of program memory. The execution of a SEL MB0 or SEL MB1 instruction within an interrupt routine is not recommended since it will not alter PC11 while in the routine, but will change the internal flip flop.

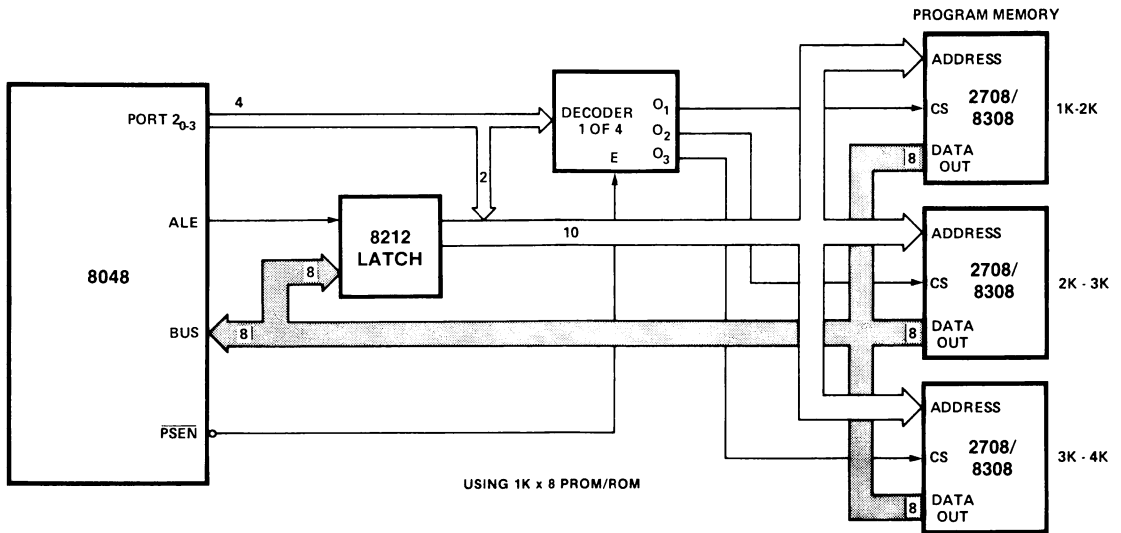
#### 3.1.3 Restoring I/O Port Information

Although the lower half of Port 2 is used to output the four most significant bits of address during an external program memory fetch, the I/O information is still outputted during certain portions of each machine cycle. I/O information is always present on Port 2 lower at the rising edge of ALE and can be sampled or latched at this time.

#### 3.1.4 Expansion Examples

The accompanying figure shows the addition of three 2708 1K X 8 EPROMs or three 8308 pin-compatible ROM replacements for a total of 4K words of program memory. The BUS port of the 8048 is connected directly to the data output lines of the memories. The lower 8 bits of address are latched in an 8212 8-bit latch using ALE as the strobe. The lower half of Port 2 provides the upper 4 bits of address and since these address bits are stable for the duration of the program memory fetch, they do not have to be latched. Two of the upper address bits are connected directly to the address inputs of the memories while the two most significant bits are decoded to provide the three chip selects needed. The PSEN output of the 8048/8748 is used to enable the chip select lines and therefore the memories.

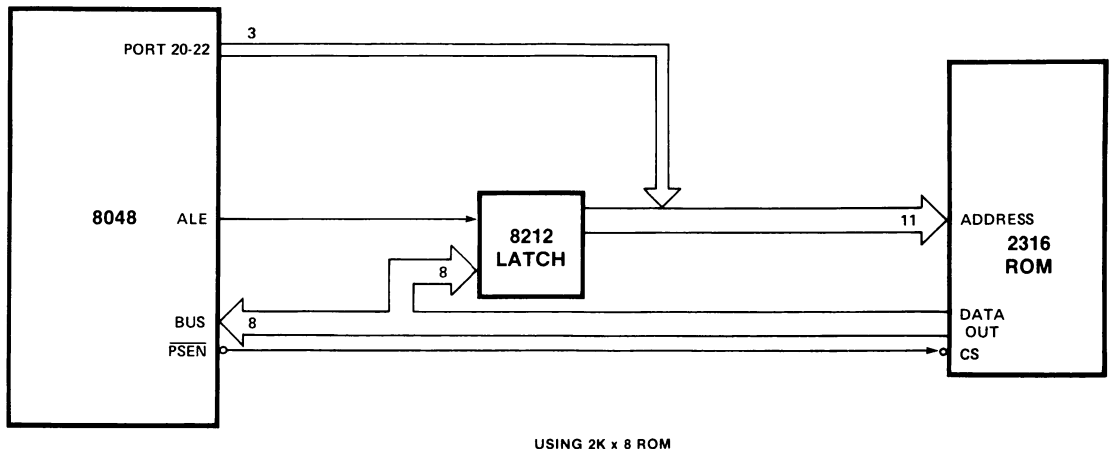
## EXPANDED MCS-48 SYSTEM



### EXPANDING MCS-48™ PROGRAM MEMORY USING STANDARD MEMORY PRODUCTS

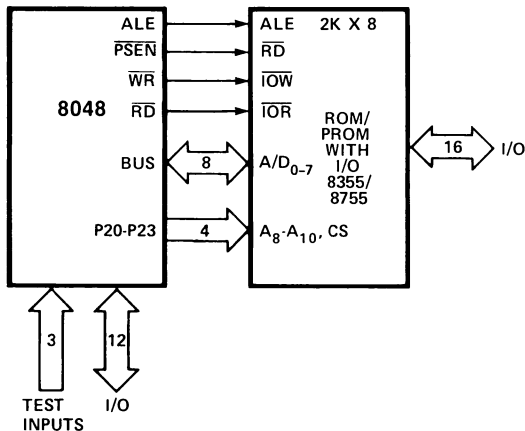
Also shown is the addition of 2K words of program memory using an 8316A 2K x 8 ROM to give a total of 3K words of program memory. In this case no chip select decoding is required and  $\overline{\text{PSEN}}$  enables the memory directly through the chip select input. If the system requires only 2K of program the same configuration can be used with an 8035 substituted for the 8048.

The next figure shows how the new 8755/8355 EPROM/ROM with I/O interfaces directly to the 8048 without the need for an address latch. The 8755/8355 contains an internal 8-bit address latch eliminating the need for an 8212 latch. In addition to a 2K X 8 program memory the 8755/8355 also contains 16 I/O lines addressable as two 8-bit ports. These ports are addressed as external RAM; there-



### EXPANDING MCS-48™ PROGRAM MEMORY USING STANDARD MEMORY PRODUCTS

fore, the  $\overline{RD}$  and  $\overline{WR}$  outputs of the 8048 are required. See the following section on data memory expansion for more detail. The subsequent section on I/O expansion explains the operation of the 16 I/O lines.



## EXTERNAL PROGRAM MEMORY INTERFACE

### 3.2 Expansion of Data Memory

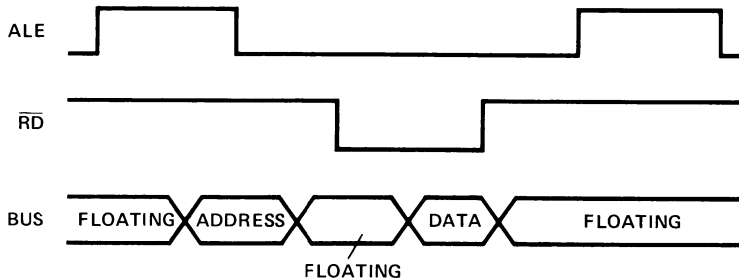
Data Memory is expanded beyond the resident 64 words by using the 8080 type bus feature of the MCS-48.

#### 3.2.1 Read/Write Cycle

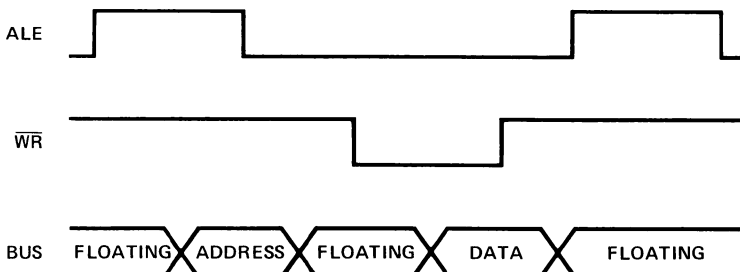
All address and data is transferred over the 8 lines of BUS. A read or write cycle occurs as follows:

1. The contents of register R0 or R1 is outputted on BUS.
2. Address Latch Enable (ALE) indicates address is valid. The trailing edge of ALE is used to latch the address externally.
3. A read ( $\overline{RD}$ ) or write ( $\overline{WR}$ ) pulse on the corresponding output pins of the 8048 indicates the type of data memory access in progress. Output data is valid at the trailing edge of  $\overline{WR}$  and input data must be valid at the trailing edge of  $\overline{RD}$ .
4. Data (8-bits) is transferred in or out over BUS.

#### READ FROM EXTERNAL DATA MEMORY



#### WRITE TO EXTERNAL DATA MEMORY



### 3.2.2 Addressing External Data Memory

External Data Memory is accessed with its own two-cycle move instructions MOVX A, @R and MOVX @R, A which transfer 8 bits of data between the accumulator and the external memory location addressed by the contents of one of the RAM Pointer Registers R0 or R1. This allows 256 locations to be addressed in addition to the resident 64 locations. Additional pages may be added by "bank switching" with extra output lines of the 8048.

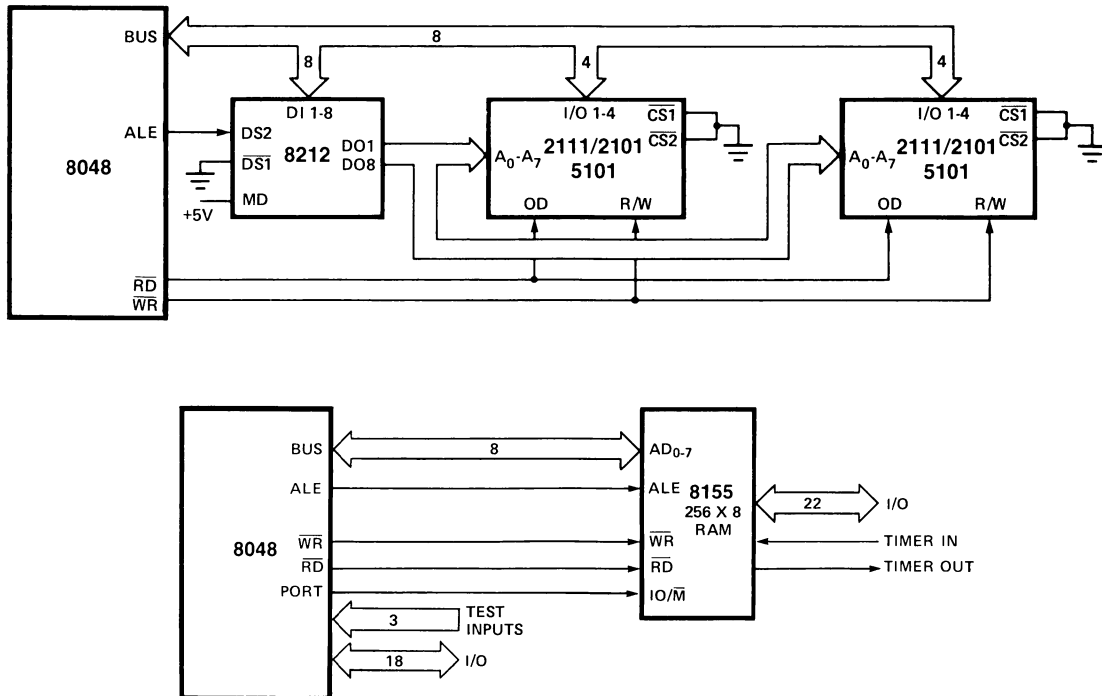
### 3.2.3 Examples of Data Memory Expansion

The accompanying figure shows how the 8048 can be expanded using standard 256 X 4 static RAMs such as the 2101-2 or its low power CMOS equivalent, the 5101. An 8212 serves as an address latch while each 4-bit half of BUS is connected directly to a bidirec-

tional 4-bit data bus of the memories. The  $\overline{WR}$  output of the processor controls the Read/Write input of the memories while the data bus output drivers of the memories are controlled by  $\overline{RD}$ . The chip select lines of the memories are continuously enabled unless additional pages of RAM are required. Also shown is the expansion of data memory using the 8155 memory and I/O expanding device. Since the 8155 has an internal 8-bit address latch it can interface directly to the 8048 without the use of an external 8212 latch. The 8155 provides an additional 256 words of static data memory and also includes 22 I/O lines and a 14 bit timer. See the following section on I/O expansion and the 8155 data sheet for more details on these additional features.

### 3.3 Expansion of Input/Output

There are three possible modes of I/O expansion with the 8048: one using a special low cost expander, the 8243; another using stan-



8048 INTERFACE TO 256 X 8 STANDARD MEMORIES

## EXPANDED MCS-48 SYSTEM

standard MCS-80 I/O devices; and a third using the combination memory/I/O expander devices the 8155, 8355, and 8755.

### 3.3.1 I/O Expander Device

The most efficient means of I/O expansion for small systems is the 8243 I/O Expander Device which requires only 4 port lines (lower half of Port 2) for communication with the 8048. The 8243 contains four 4-bit I/O ports which serve as extension of the on chip I/O and are addressed as ports #4-7. The following operations may be performed on these ports:

1. Transfer Accumulator to Port.
2. Transfer Port to Accumulator.
3. AND Accumulator to Port.
4. OR Accumulator to Port.

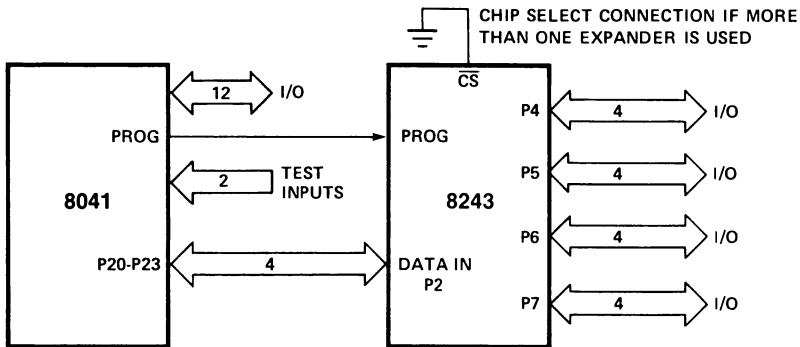
A 4-bit transfer from a port to the lower half of the Accumulator sets the most significant four

bits to zero. All communication between the 8048 and the 8243 occurs over Port 2 lower (P20-P23) with timing provided by an output pulse on the PROG pin of the processor. Each transfer consists of two 4-bit nibbles:

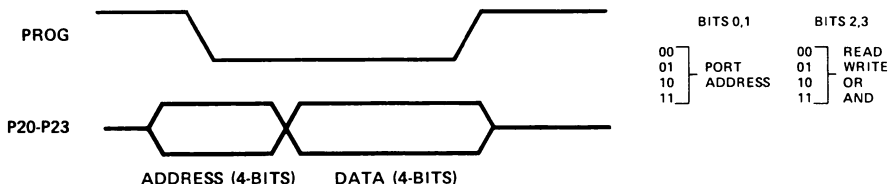
The first containing the "op code" and port address and the second containing the actual 4 bits of data.

Nibble 1				Nibble 2			
3	2	1	0	3	2	1	0
I	I	A	A	d	d	d	d
Port Address				Instruction Code			
AA				II			
00—Port #4				00 Read			
01—Port #5				01 Write			
10—Port #6				10 OR			
11—Port #7				11 AND			

### EXPANDER INTERFACE



### OUTPUT EXPANDER TIMING



A high to low transition of the PROG line indicates that address is present while a low to high transition indicates the presence of data. Additional 8243's may be added to the four bit bus and chip selected using additional output lines from the 8048/8748.

### I/O Port Characteristics

Each of the four 4-bit ports of the 8243 can serve as either input or output and can provide high drive capability in both the high and low state.

### 3.3.2 I/O Expansion with Standard Peripherals

Standard 8080 type I/O devices may be added to the MCS-48 using the same bus and timing used for Data Memory expansion. I/O devices reside on the Data Memory bus and in the data memory address space and are accessed with the same MOVX instructions. See the previous section on data memory expansion for a description of the timing. The following is a few of the Standard MCS-80 devices which are very useful in MCS-48 systems:

- 8214 Priority Interrupt Encoder
- 8251 Serial Communications Interface
- 8255 General Purpose Programmable I/O
- 8279 Keyboard/Display Interface
- 8253 Interval Timer

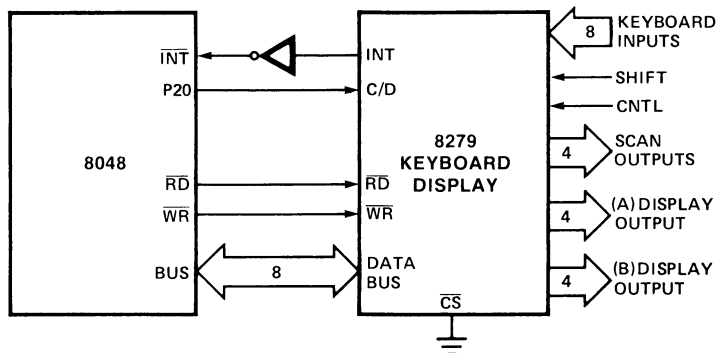
See Chapter 7 for detailed data sheets on these components.

### 3.3.3 Combination Memory and I/O Expanders

As mentioned in the sections on program and data memory expansion the 8355/8755 and 8155 expanders also contain I/O capability.

**8355/8755:** These two parts are ROM and EPROM equivalents and therefore contain the same I/O structure. I/O consists of two 8-bit ports which normally reside in the external data memory address space and are accessed with MOVX instructions. Associated with each port is an 8-bit Data Direction Register which defines each bit in the port as either an input or an output. The data direction registers are directly addressable thereby allowing the user to define under software control each individual bit of the ports as either input or output. All outputs are statically latched and double buffered. Inputs are not latched.

**8155:** I/O on the 8155 is configured as two 8-bit programmable I/O ports and one 6-bit programmable port. These three registers and a Control/Status register are accessible as external data memory with the MOVX instructions. The contents of the control register determines the mode of the three ports. The ports can be programmed as input or output with or without associated handshake communication lines. In the handshake mode, lines of the six-bit port become input and output strobes for the two 8-bit ports. See the



KEYBOARD/DISPLAY INTERFACE

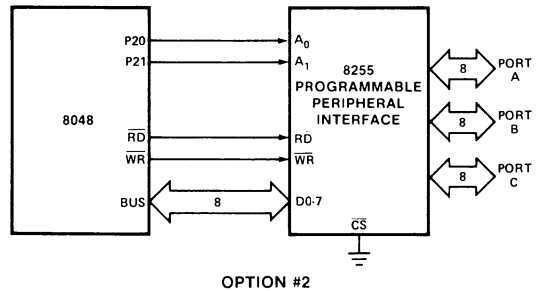
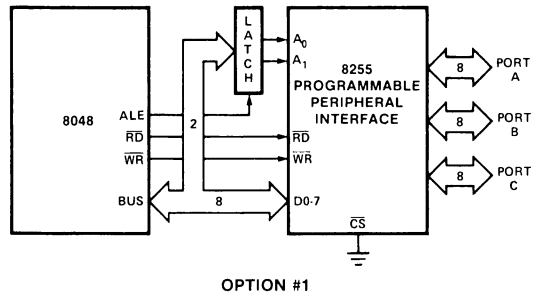
data sheet in the Chapter 6 for details. Also included in the 8155 is a 14-bit programmable timer. The clock input to the timer and the timer overflow output are available on external pins. The timer can be programmed to stop on terminal count or to continuously re-load itself. A square wave or pulse output on terminal count can also be specified.

## I/O Expansion Examples

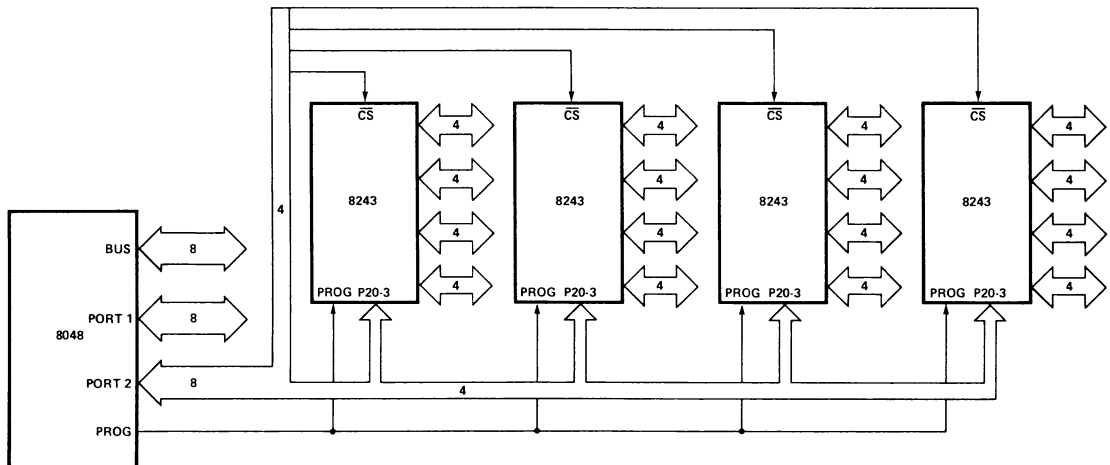
The accompanying figure shows the expansion of I/O using multiple 8243's. The only difference from a single 8243 system is the addition of chip selects provided by additional 8048 output lines. Two output lines and two inverters could also be used to address the four chips. Large numbers of 8243's would require a chip select decoder chip such as the 8205 to save I/O pins.

Also shown is the 8048 interface to a standard MCS-80 peripheral; in this case, the 8255 Programmable Peripheral Interface, a 40 pin part which provides three 8-bit programmable I/O ports. The 8255 bus interface is typical of programmable MCS-80 peripherals with an 8-bit bidirectional data bus, a  $\overline{RD}$  and  $\overline{WR}$  input for Read/Write control, a  $\overline{CS}$

(chip select) input used to enable the Read/Write control logic and the address inputs used to select various internal registers.



## INTERFACE TO MCS 80 PERIPHERALS



## LOW COST I/O EXPANSION

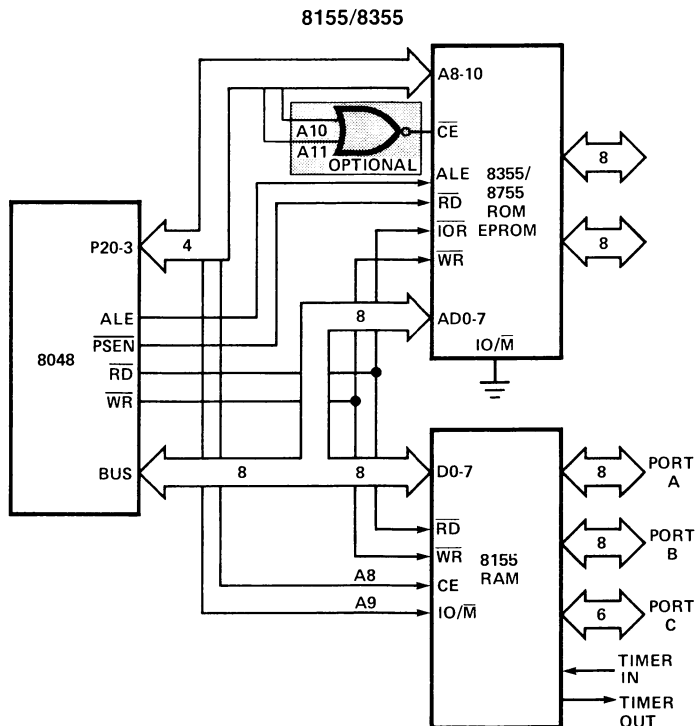


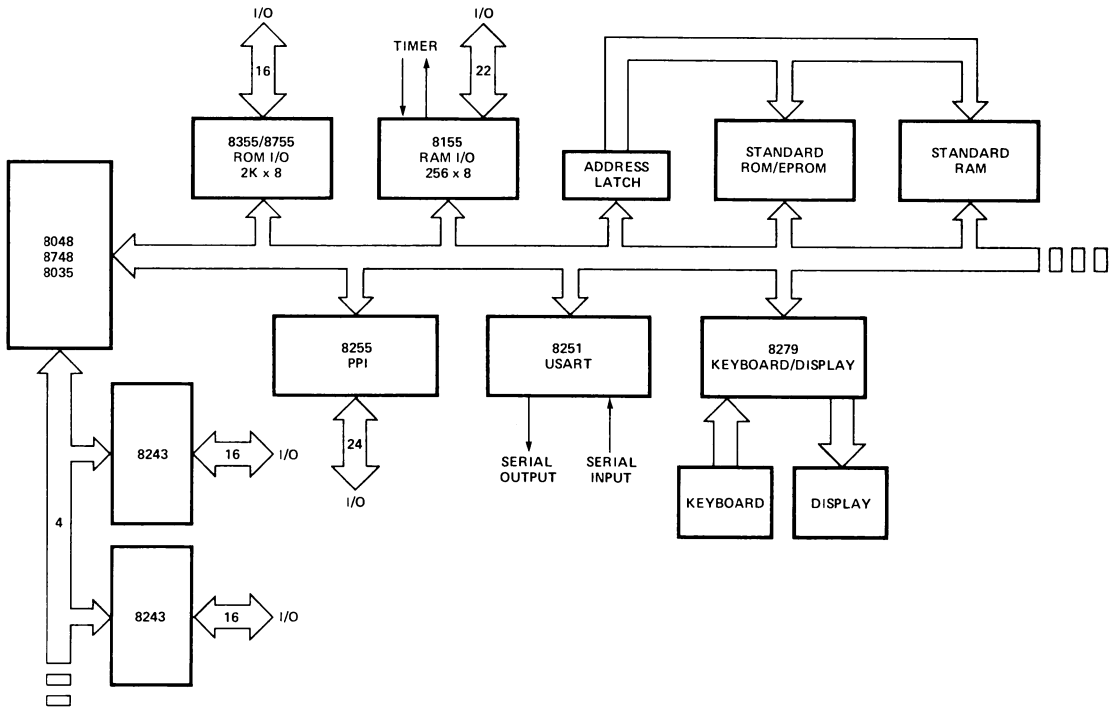
interconnection to the 8048 is very straightforward with  $\overline{\text{BUS}}$ ,  $\overline{\text{RD}}$ , and  $\overline{\text{WR}}$  connecting directly to the corresponding pins on the 8255. The only design consideration is the way in which the internal registers of the 8255 are to be addressed. If the registers are to be addressed as external data memory using the MOVX instructions, the appropriate number of address bits (in this case, 2) must be latched on  $\overline{\text{BUS}}$  using ALE as described in the section on external data memories. If only a single device is connected to  $\overline{\text{BUS}}$ , the 8255 may be continuously selected by grounding  $\overline{\text{CS}}$ . If multiple 8255's are used, additional address bits can be latched and used as chip selects.

A second addressing method eliminates external latches and chip select decoders by using output port lines as address and chip select lines directly. This method, of course, requires the setting of an output port with address information prior to executing a MOVX instruction.

### 3.4 Multi-Chip MCS-48 Systems

The accompanying figure shows the addition of two memory expanders to the 8048, one 8355/8755 ROM and one 8155 RAM. The main consideration in designing such a system is the addressing of the various memories and I/O ports. Note that in this configuration address lines  $A_{10}$  and  $A_{11}$  have been ORed to chip select the 8355. This ensures that the chip is active for all external program memory fetches in the 1K to 3K range and is disabled for all other addresses. This gating has been added to allow the I/O port of the 8355 to be used. If the chip was left selected all the time there would be conflict between these ports and the RAM and I/O of the 8155. The NOR gate could be eliminated and  $A_{11}$  connected directly to the  $\overline{\text{CE}}$  (instead of  $\overline{\text{CE}}$ ) input of the 8355; however, this would create a 1K word "hole" in the program memory by causing the 8355 to be active in the 2K to 4K range instead of the normal 1K to 3K range.





## MCS-48 EXPANSION CAPABILITY

In this system the various locations are addressed as follows:

**Data RAM**—Addresses 0 to 255 when Port 2 Bit 0 has been previously set = 1 and Bit 1 set = 0

**RAM I/O**—Addresses 0 to 3 when Port 2 Bit 0 = 1 and Bit 1 = 1

**ROM I/O**—Addresses 0 to 3 when Port 2 Bit 2 or Bit 3 = 1

### 3.5 Bank Switching

Certain systems may require more than the 4K words of program memory which are directly addressable by the program counter or more than the 256 data memory and I/O locations directly addressable by the pointer

registers R0 and R1. These systems can be achieved using “bank switching” techniques. Bank switching is merely the selection of various blocks or “banks” of memory using dedicated output port lines from the processor. In the case of the 8048 program memory is selected in blocks of 4K words at a time while data memory and I/O are enabled 256 words at a time.

The most important consideration in implementing two or more banks is the software required to cross the bank boundaries. Each crossing of the boundary requires that the processor first write a control bit to an output port before accessing memory or I/O in the new bank. If program memory is being switched, programs should be organized to

keep boundary crossings to a minimum. Jumping to subroutines across the boundary should be avoided when possible since the programmer must keep track of which bank to return to after completion of the subroutine. If these subroutines are to be nested and accessed from either bank, a software "stack" should be implemented to save the bank

switch bit just as if it were another bit of the program counter.

From a hardware standpoint bank switching is very straight-forward and involves only the connection of an I/O line or lines as bank enable signals. These enables are ANDed with normal memory and I/O chip select signals to activate the proper bank.

1. The first step in the process of the investigation is the identification of the problem. This is done by the investigator who is responsible for the study. The investigator must first identify the problem and then determine the scope of the problem. The scope of the problem is determined by the investigator who is responsible for the study. The investigator must first identify the problem and then determine the scope of the problem. The scope of the problem is determined by the investigator who is responsible for the study.

## Chapter 4

# INSTRUCTION SET



## INSTRUCTION SET

4.0 Introduction .....	4-1
4.1 Instruction Set Description .....	4-4

# INSTRUCTION SET

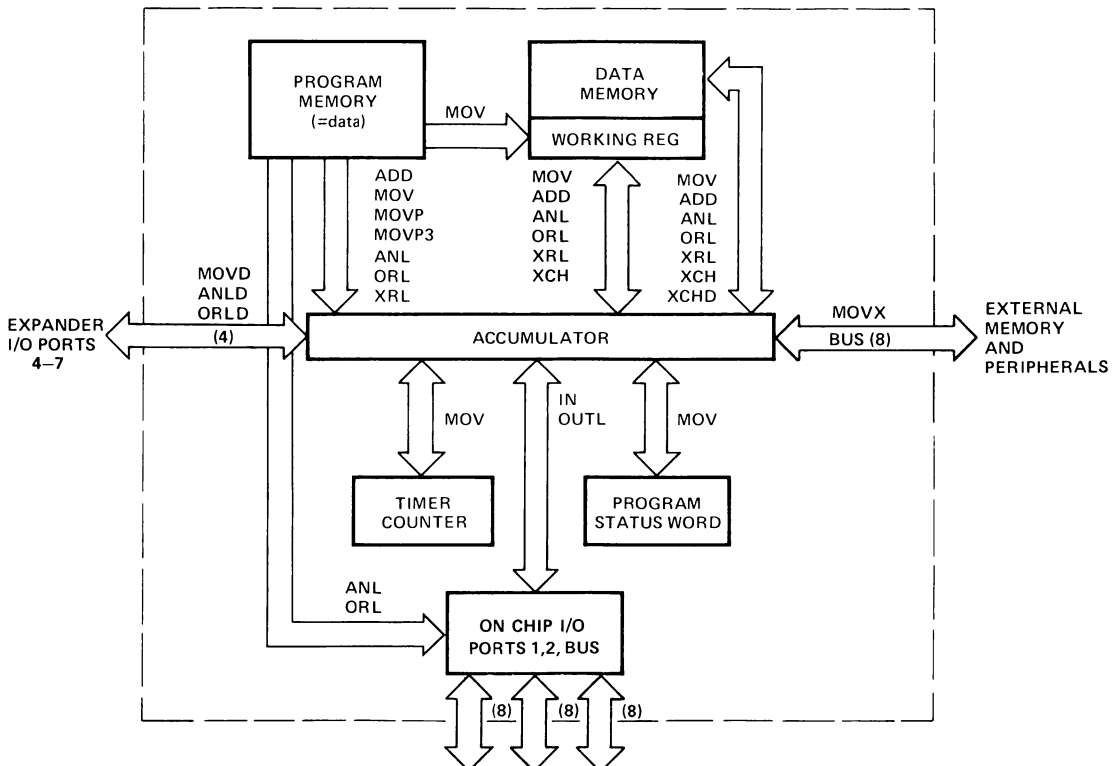
## 4.0 INTRODUCTION

The MCS-48 instruction set is extensive for a machine of its size and has been tailored to be straightforward and very efficient in its use of program memory. All instructions are either one or two bytes in length and over 70% are only one byte long. Also, all instructions execute in either one or two cycles (2.5 sec or 5.0 sec when using a 6 MHz XTAL) and over 50% of all instructions execute in a single cycle. Double cycle instructions include all immediate instructions, and all I/O instructions.

The MCS-48 microcomputers have been designed to efficiently handle arithmetic operations in both binary and BCD as well as to efficiently handle the single bit operations required in control applications. Special instructions have also been included to simplify loop counters, table lookup routines, and N-way branch routines.

### Data Transfers

As can be seen in the accompanying diagram, the 8-bit accumulator is the central



## DATA TRANSFER INSTRUCTIONS

point for all data transfers within the 8048. Data can be transferred between the 8 registers of each working register bank and the accumulator directly, i.e. the source or destination register is specified by the instruction. The remaining locations of the internal RAM array are referred to as Data Memory and are addressed indirectly via an address stored in either R0 or R1 of the active working register bank. R0 and R1 are also used to indirectly address external data memory when it is present. Transfers to and from internal RAM require one cycle while transfers to external RAM require two. Constants stored in Program Memory can be loaded directly to the accumulator and to the 8 working registers. Data can also be transferred directly between the accumulator and the on-board timer/counter or the accumulator and the Program Status word (PSW). Writing to the PSW alters machine status accordingly and provides a means of restoring status after an interrupt or of altering the stack pointer if necessary.

### Accumulator Operations

Immediate data, data memory, or the working registers can be added with or without carry to the accumulator. These sources can also be ANDed, ORed, or Exclusive ORed to the accumulator. Data may be moved to or from the accumulator and working registers or data memory. The two values can also be exchanged in a single operation.

In addition, the lower 4 bits of the accumulator can be exchanged with the lower 4-bits of any of the internal RAM locations. This instruction, along with an instruction which swaps the upper and lower 4-bit halves of the accumulator, provides for easy handling of 4-bit quantities, including BCD numbers. To facilitate BCD arithmetic, a Decimal Adjust instruction is included. This instruction is used to correct the result of the binary addition of two two-digit BCD numbers. Performing a decimal adjust on the result in the accumulator produces the required BCD result.

Finally, the accumulator can be: incremented, decremented, cleared, or complemented and can be rotated left or right 1-bit at a time with or without carry.

Although there is no subtract instruction in the 8048, this operation can be easily implemented with three single-byte single-cycle instructions.

A value may be subtracted from the accumulator with the result in the accumulator by:

- Complementing the accumulator
- Adding the value to the accumulator
- Complementing the accumulator.

### Register Operations

The working registers can be accessed via the accumulator as explained above, or can be loaded immediate with constraints from program memory. In addition, they can be incremented or decremented or used as loop counters using the decrement and skip, if not zero instruction, as explained under branch instructions.

All Data Memory including working registers can be accessed with indirect instructions via R0 and R1 and can be incremented.

### Flags

There are four user accessible flags in the 8048: Carry, Auxillary Carry, F0, and F1. Carry indicates overflow of the accumulator, and Auxillary Carry is used to indicate overflow between BCD digits and is used during decimal adjust operation. Both Carry and Auxillary Carry are accessible as part of the program status word and are stored on the stack during subroutines. F0 and F1 are undedicated general purpose flags to be used as the programmer desires. Both flags can be cleared or complemented and tested by conditional jump instructions. F0 is also accessible via the Program Status word and is stored on the stack with the carry flags.

### Branch Instructions

The unconditional jump instruction is two bytes and allows jumps anywhere in the first



2K words of program memory. Jumps to the second 2K of memory (4K words are directly addressable) are made by first executing a select memory bank instruction then executing the jump instruction. The 2K boundary can only be crossed via a jump or subroutine call instruction i.e. the bank switch does not occur until a jump is executed. Once a memory bank has been selected all subsequent jumps will be to the selected bank until another select memory bank instruction is executed. A subroutine in the opposite bank can be accessed by a select memory bank instruction followed by a call instruction. Upon completion of the subroutine execution will automatically return to the original bank; however, unless the original bank is reselected, the next jump instruction encountered will again transfer execution to the opposite bank.

Conditional jumps can test the following inputs and machine status:

- T0 Input pin
- T1 Input pin
- $\overline{\text{INT}}$  Input pin
- Accumulator Zero
- Any bit of Accumulator
- Carry Flag
- F0 Flag
- F1 Flag

Conditional jumps allow a branch to any address within the current page (256 words) of execution. The conditions tested are the instantaneous values at the time the conditional jump is executed. For instance, the jump on accumulator zero instruction tests the accumulator itself not an intermediate zero flag.

The decrement register and skip if not zero instruction combines a decrement and a branch instruction to create an instruction very useful in implementing a loop counter. This instruction can designate any one of the 8 working registers as a counter and can effect a branch to any address within the current page of execution.

A single byte indirect jump instruction allows the program to be vectored to any one of

several different locations based on the contents of the accumulator. The contents of the accumulator points to a location in program memory which contains the jump address. The 8-bit jump address refers to the current page of execution. This instruction could be used, for instance, to vector to any one of several routines based on an ASCII character which has been loaded in the accumulator. In this way ASCII key inputs can be used to initiate various routines.

### Subroutines

Subroutines are entered by executing a call instruction. Calls can be made like unconditional jumps to any address in a 2K word bank and jumps across the 2K boundary are executed in the same manner. Two separate return instructions determine whether or not status (upper 4-bits of PSW) is restored upon return from the subroutine.

The return and restore status instruction also signals the end of an interrupt service routine if one has been in progress.

### Timer Instructions

The 8-bit on board timer/counter can be loaded or read via the accumulator while the counter is stopped or while counting. The counter can be started as a timer with an internal clock source or as an event counter or timer with an external clock applied to the T1 input pin. The instruction executed determines which clock source is used. A single instruction stops the counter whether it is operating with an internal or an external clock source. In addition, two instructions allow the timer interrupt to be enabled or disabled.

### Control Instructions

Two instructions allow the external interrupt source to be enabled or disabled. Interrupts are initially disabled and are automatically disabled while an interrupt service routine is in progress and re-enabled afterward.

There are four memory bank select instructions, two to designate the active working register bank and two to control program

memory banks. The operation of the program memory bank switch is explained in section 3.1.2. The working register bank switch instructions allow the programmer to immediately substitute a second 8 register working register bank for the one in use. This effectively provides 16 working registers or it can be used as a means of quickly saving the contents of the registers in response to an interrupt. The user has the option to switch or not to switch banks on interrupt. However, if the banks are switched, the original bank will be automatically restored upon execution of a return and restore status instruction at the end of the interrupt service routine.

A special instruction enables an internal clock, which is the XTAL frequency divided by three, to be output on pin T0. This clock can be used as a general purpose clock in the users system. This instruction should be used only to initialize the system since the clock output can be disabled only by application of system reset.

### Input/Output Instructions

Ports 1 and 2 are 8-bit static I/O ports which can be loaded to and from the accumulator. Outputs are statically latched but inputs are not latched and must be read while inputs are present. In addition, immediate data from program memory can be ANDed or ORed directly to Port 1 and Port 2 with the result remaining on the port. This allows "masks" stored in program memory to selectively set or reset individual bits of the I/O ports. Ports 1 and 2 are configured to allow input on a given pin by first writing a "1" out to the pin.

An 8-bit port called BUS can also be accessed via the accumulator and can have statically latched outputs as well. It too can have immediate data ANDed or ORed directly to its outputs, however, unlike ports 1 and 2, all eight lines of BUS must be treated as either input or output at any one time. In addition to being a static port, BUS can be used as a true synchronous bi-directional port using the Move External instructions used to access external data memory. When these instructions are executed a cor-

responding READ or WRITE pulse is generated and data is valid only at that time. When data is not being transferred BUS is in a high impedance state.

The basic three on board I/O ports can be expanded via a 4-bit expander bus using half of port 2. I/O expander devices on this bus consist of four 4-bit ports which are addressed as ports 4 through 7. These ports have their own AND and OR instructions like the on board ports as well as move instructions to transfer data in or out. The expander AND and OR instructions, however, combine the contents of accumulator with the selected port rather than immediate data as is done with the on board ports.

I/O devices can also be added externally using the BUS port as the expansion bus. In this case the I/O ports become "memory mapped", i.e. they are addressed in the same way as external data memory and exist in the external data memory address space addressed by pointer register R0 or R1.

### 4.1 Instruction Set Description

The following pages describe the MCS-48 instruction set in detail. The instruction set is first summarized with instructions grouped functionally. This summary page is followed by a detailed description listed alphabetically by mnemonic opcode.

The alphabetical listing includes the following information:

- Mnemonic
- Machine Code
- Verbal Description
- Symbolic Description
- Assembly Language Example

The machine code is represented with the most significant bit (7) to the left and two byte instructions are represented with the first byte on the left. The assembly language examples are formulated as follows:

Arbitrary  
Label: Mnemonic, Operand; Descriptive Comment  
See section 1.2.2 for a description and example of an assembly language program.

# INSTRUCTION SET SUMMARY

	Mnemonic	Description	Bytes	Cycle		Mnemonic	Description	Bytes	Cycles
Accumulator	ADD A, R	Add register to A	1	1	Subroutine	CALL	Jump to subroutine	2	2
	ADD A, @R	Add data memory to A	1	1		RET	Return	1	2
	ADD A, #data	Add immediate to A	2	2		RETR	Return and restore status	1	2
	ADDC A, R	Add register with carry	1	1	Flags	CLR C	Clear Carry	1	1
	ADDC A, @R	Add data memory with carry	1	1		CPL C	Complement Carry	1	1
	ADDC A, #data	Add immediate with carry	2	2		CLR F0	Clear Flag 0	1	1
	ANL A, R	And register to A	1	1		CPL F0	Complement Flag 0	1	1
	ANL A, @R	And data memory to A	1	1		CLR F1	Clear Flag 1	1	1
	ANL A, #data	And immediate to A	2	2		CPL F1	Complement Flag 1	1	1
	ORL A, R	Or register to A	1	1	Data Moves	MOV A, R	Move register to A	1	1
	ORL A, @R	Or data memory to A	1	1		MOV A, @R	Move data memory to A	1	1
	ORL A, #data	Or immediate to A	2	2		MOV A, #data	Move immediate to A	2	2
	XRL A, R	Exclusive Or register to A	1	1		MOV R, A	Move A to register	1	1
	XRL A, @R	Exclusive or data memory to A	1	1		MOV @R, A	Move A to data memory	1	1
	XRL A, #data	Exclusive or immediate to A	2	2		MOV R, #data	Move immediate to register	2	2
	INC A	Increment A	1	1		MOV @R, #data	Move immediate to data memory	2	2
	DEC A	Decrement A	1	1		MOV A, PSW	Move PSW to A	1	1
	CLR A	Clear A	1	1		MOV PSW, A	Move A to PSW	1	1
	CPL A	Complement A	1	1		XCH A, R	Exchange A and register	1	1
	DA A	Decimal Adjust A	1	1		XCH A, @R	Exchange A and data memory	1	1
	SWAP A	Swap nibbles of A	1	1		XCHD A, @R	Exchange nibble of A and register	1	1
	RL A	Rotate A left	1	1		MOVX A, @R	Move external data memory to A	1	2
	RLC A	Rotate A left through carry	1	1		MOVX @R, A	Move A to external data memory	1	2
	RR A	Rotate A right	1	1		MOVP A, @A	Move to A from current page	1	2
	RRC A	Rotate A right through carry	1	1		MOVP3 A, @A	Move to A from Page 3	1	2
Input/Output	IN A, P	Input port to A	1	2	Timer/Counter	MOV A, T	Read Timer/Counter	1	1
	OUTL P, A	Output A to port	1	2		MOV T, A	Load Timer/Counter	1	1
	ANL P, #data	And immediate to port	2	2		STRT T	Start Timer	1	1
	ORL P, #data	Or immediate to port	2	2		STRT CNT	Start Counter	1	1
	INS A, BUS	Input BUS to A	1	2		STOP TCNT	Stop Timer/Counter	1	1
	OUTL BUS, A	Output A to BUS	1	2		EN TCNTI	Enable Timer/Counter Interrupt	1	1
	ANL BUS, #data	And immediate to BUS	2	2		DIS TCNTI	Disable Timer/Counter Interrupt	1	1
	ORL BUS, #data	Or immediate to BUS	2	2	Control	EN I	Enable external interrupt	1	1
	MOVD A, P	Input Expander port to A	1	2		DIS I	Disable external interrupt	1	1
	MOVD P, A	Output A to Expander port	1	2		SEL RB0	Select register bank 0	1	1
Registers	ANLD P, A	And A to Expander port	1	2		SEL RB1	Select register bank 1	1	1
	ORLD P, A	Or A to Expander port	1	2		SEL MB0	Select memory bank 0	1	1
Branch	INC R	Increment register	1	1		SEL MB1	Select memory bank 1	1	1
	INC @R	Increment data memory	1	1		ENTO CLK	Enable Clock output on T0	1	1
	DEC R	Decrement register	1	1	NOP	NOP	No Operation	1	1
	JMP addr	Jump unconditional	2	2					
	JMPP @A	Jump indirect	1	2					
	DJNZ R, addr	Decrement register and skip	2	2					
	JC addr	Jump on Carry = 1	2	2					
	JNC addr	Jump on Carry = 0	2	2					
	JZ addr	Jump on A Zero	2	2					
	JNZ addr	Jump on A not Zero	2	2					
	JT0 addr	Jump on T0 = 1	2	2					
	JNT0 addr	Jump on T0 = 0	2	2					
	JT1 addr	Jump on T1 = 1	2	2					
	JNT1 addr	Jump on T1 = 0	2	2					
	JF0 addr	Jump on F0 = 1	2	2					
	JF1 addr	Jump on F1 = 1	2	2					
	JTF addr	Jump on timer flag	2	2					
	JNI addr	Jump on INT = 0	2	2					
	JBB addr	Jump on Accumulator Bit	2	2					

# MCS-48™ INSTRUCTION SET

## SYMBOLS AND ABBREVIATIONS USED

A	Accumulator
AC	Auxillary Carry
addr	12-Bit Program Memory Address
Bb	Bit Designator (b=0-7)
BS	Bank Switch
BUS	BUS Port
C	Carry
CLK	Clock
CNT	Event Counter
D	Mnemonic for 4-Bit Digit (Nibble)
data	8-Bit Number or Expression
DBF	Memory Bank Flip-Flop
F0, F1	Flag 0, Flag 1
I	Interrupt
P	Mnemonic for “in-page” Operation
PC	Program Counter
Pp	Port Designator (p=1, 2 or 4-7)
PSW	Program Status Word
Rr	Register Designator (r=0, 1 or 0-7)
SP	Stack Pointer
T	Timer
TF	Timer Flag
T0, T1	Test 0, Test 1
X	Mnemonic for External RAM
#	Immediate Data Prefix
@	Indirect Address Prefix
\$	Current Value of Program Counter
(X)	Contents of X
((X))	Contents of Location Addressed by X
←	Is Replaced by

**ADD A,R<sub>r</sub> Add Register Contents to Accumulator**

---

0 1 1 0	1 r r r
---------	---------

The contents of register 'r' are added to the accumulator. Carry is affected.

$$(A) \leftarrow (A) + (R_r) \quad r=0-7$$

**Example:** ADDREG: ADD A,R6 ;ADD REG 6 CONTENTS  
;TO ACC

**ADD A,@R<sub>r</sub> Add Data Memory Contents to Accumulator**

---

0 1 1 0	0 0 0 r
---------	---------

The contents of the resident data memory location addressed by register 'r' bits 0-5 are added to the accumulator. Carry is affected.

$$(A) \leftarrow (A) + ((R_r)) \quad r=0-1$$

**Example:** ADDM: MOV R0, #0AFH ;MOVE 'AF' HEX TO REG 0  
ADD A, @R0 ;ADD VALUE OF LOCATION  
;47 TO ACC

**ADD A,#data Add Immediate Data to Accumulator**

---

0 0 0 0	0 0 1 1	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. The specified data is added to the accumulator. Carry is affected

$$(A) \leftarrow (A) + \text{data}$$

**Example:** ADDID: ADD A,#ADDER: ;ADD VALUE OF SYMBOL  
;'ADDER' TO ACC

**ADDC A,R<sub>r</sub> Add Carry and Register Contents to Accumulator**

---

0 1 1 1	1 r r r
---------	---------

The content of the carry bit is added to accumulator location 0 and the carry bit cleared. The contents of register 'r' are then added to the accumulator. Carry is affected.

$$(A) \leftarrow (A) + (R_r) + (C) \quad r=0-7$$

**Example:** ADDRGC: ADDC A,R4 ;ADD CARRY AND REG 4  
;CONTENTS TO ACC

**ADDC A,@R<sub>r</sub> Add Carry and Data Memory Contents to Accumulator**

0	1	1	1	0	0	0	r
---	---	---	---	---	---	---	---

The content of the carry bit is added to accumulator location 0 and the carry bit cleared. Then the contents of the resident data memory location addressed by register 'r' bits 0-5 are added to the accumulator. Carry is affected.

$$(A) \leftarrow (A) + ((Rr)) + (C) \quad r=0-1$$

**Example:** ADDMC: MOV R1,#40 ;MOVE '40' DEC TO REG 1  
 ADDC A,@R1 ;ADD CARRY AND LOCATION 40  
 ;CONTENTS TO ACC

**ADDC A,#data Add Carry and Immediate Data to Accumulator**

0	0	0	1	0	0	1	1	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
---	---	---	---	---	---	---	---	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

This is a 2-cycle instruction. The content of the carry bit is added to accumulator location 0 and the carry bit cleared. Then the specified data is added to the accumulator. Carry is affected.

$$(A) \leftarrow (A) + \text{data} + (C)$$

**Example:** ADDC A,#225 ;ADD CARRY AND '225' DEC  
 ;TO ACC

**ANL A,R<sub>r</sub> Logical AND Accumulator With Register Mask**

0	1	0	1	1	r	r	r
---	---	---	---	---	---	---	---

Data in the accumulator is logically ANDed with the mask contained in working register 'r'.

$$(A) \leftarrow (A) \text{ AND } (Rr) \quad r=0-7$$

**Example:** ANDREG: ANL A,R3 ;'AND' ACC CONTENTS WITH MASK  
 ;IN REG 3

**ANL A,@R<sub>r</sub> Logical AND Accumulator With Memory Mask**

0	1	0	1	0	0	0	r
---	---	---	---	---	---	---	---

Data in the accumulator is logically ANDed with the mask contained in the data memory location referenced by register 'r', bits 0-5.

$$(A) \leftarrow (A) \text{ AND } ((Rr)) \quad r=0-1$$

**Example:** ANDDM: MOV R0,#0FFH ;MOVE 'FF' HEX TO REG 0  
 ANL A, @R0 ;'AND' ACC CONTENTS WITH  
 ;MASK IN LOCATION 63

**ANL A,#data Logical AND Accumulator With Immediate Mask**

---

0 1 0 1	0 0 1 1	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data in the accumulator is logically ANDed with an immediately-specified mask.

(A) ← (A) AND data

**Examples:** ANDID: ANL A,#OAFH ;'AND' ACC CONTENTS  
;WITH MASK 10101111  
ANL A,#3+X/Y ;'AND' ACC CONTENTS  
;WITH VALUE OF EXP  
;'3+X/Y'

**ANL BUS,#data Logical AND BUS With Immediate Mask**

---

1 0 0 1	1 0 0 0	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data on the BUS port is logically ANDed with an immediately-specified mask. This instruction assumes prior specification of an 'OUTL BUS, A' instruction.

(BUS) ← (BUS) AND data

**Example:** ANDBUS: ANL BUS, #MASK ;'AND' BUS CONTENTS  
;WITH MASK EQUAL VALUE  
;OF SYMBOL 'MASK'

**ANL Pp,#data Logical AND Port 1-2 With Immediate Mask**

---

1 0 0 1	1 0 p p	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data on port 'p' is logically ANDed with an immediately-specified mask.

(Pp) ← (Pp) AND data p=1-2

**Example:** ANDP2: ANL P2,#0F0H ;'AND' PORT 2 CONTENTS  
;WITH MASK 'F0' HEX  
;(CLEAR P20-23)

**ANLD Pp,A Logical AND Port 4-7 With Accumulator Mask**

---

1 0 0 1	1 1 p p
---------	---------

This is a 2-cycle instruction. Data on port 'p' is logically ANDed with the digit mask contained in accumulator bits 0-3.

(Pp) ← (Pp) AND (A0-3) p=4-7

## INSTRUCTION SET

Note: The mapping of port 'p' to opcode bits 0-1 is as follows:

1 0	Port
0 0	4
0 1	5
1 0	6
1 1	7

**Example:** ANDP4: ANLD P4,A ;'AND' PORT 4 CONTENTS  
;WITH ACC BITS 0-3

### CALL address Subroutine Call

a <sub>10</sub> a <sub>9</sub> a <sub>8</sub> 1	0 1 0 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---	---------	---	---

This is a 2-cycle instruction. The program counter and PSW bits 4-7 are saved in the stack. The stack pointer (PSW bits 0-2) is updated. Program control is then passed to the location specified by 'address'. PC bit 11 is determined by the most recent SEL MB instruction.

Execution continues at the instruction following the CALL upon return from the subroutine.

((SP)) — (PC), (PSW 4-7)  
(SP) — (SP)+1  
(PC<sub>8-10</sub>) — (addr<sub>8-10</sub>)  
(PC<sub>0-7</sub>) — addr<sub>0-7</sub>  
(PC<sub>11</sub>) — DBF

**Example:** Add three groups of two numbers. Put subtotals in locations 50, 51 and total in location 52.

```

MOV R0,#50 ;MOVE '50' DEC TO ADDRESS
;REG 0
BEGADD: MOV A,R1 ;MOVE CONTENTS OF REG 1
;TO ACC
ADD A,R2 ;ADD REG 2 TO ACC
CALL SUBTOT;CALL SUBROUTINE 'SUBTOT'
ADD A,R3 ;ADD REG 3 TO ACC
ADD A,R4 ;ADD REG 4 TO ACC
CALL SUBTOT;CALL SUBROUTINE 'SUBTOT'
ADD A,R5 ;ADD REG 5 TO ACC
ADD A,R6 ;ADD REG 6 TO ACC
CALL SUBTOT;CALL SUBROUTINE 'SUBTOT'

SUBTOT: MOV @R0,A ;MOVE CONTENTS OF ACC TO
;LOCATION ADDRESSED BY
;REG 0
INC R0 ;INCREMENT REG 0
RET ;RETURN TO MAIN PROGRAM

```



**CLR A Clear Accumulator**

0 0 1 0	0 1 1 1
---------	---------

The contents of the accumulator are cleared to zero.

$A \leftarrow 0$

**CLR C Clear Carry Bit**

1 0 0 1	0 1 1 1
---------	---------

During normal program execution, the carry bit can be set to one by the ADD, ADDC, RLC, CPL C, RRC, and DAA instructions. This instruction resets the carry bit to zero.

$C \leftarrow 0$

**CLR F1 Clear Flag 1**

1 0 1 0	0 1 0 1
---------	---------

Flag 1 is cleared to zero.

$(F1) \leftarrow 0$

**CLR F0 Clear Flag 0**

1 0 0 0	0 1 0 1
---------	---------

Flag 0 is cleared to zero.

$(F0) \leftarrow 0$

**CPL A Complement Accumulator**

0 0 1 1	0 1 1 1
---------	---------

The contents of the accumulator are complemented. This is strictly a one's complement. Each one is changed to zero and vice-versa.

$(A) \leftarrow \text{NOT } (A)$

**Example:** Assume accumulator contains 01101010.

CPLA: CPL A ;ACC CONTENTS ARE COMPLEMENTED TO 10010101

**CPL C Complement Carry Bit**

1 0 1 0	0 1 1 1
---------	---------

The setting of the carry bit is complemented; one is changed to zero, and zero is changed to one.

$(C) \leftarrow \text{NOT } (C)$

**Example:** Set C to one; current setting is unknown.

CTO1: CLR C ;C IS CLEARED TO ZERO  
CPL C ;C IS SET TO ONE

**CPL F0 Complement Flag 0**

1 0 0 1	0 1 0 1
---------	---------

The setting of flag 0 is complemented; one is changed to zero, and zero is changed to one.

$F0 \leftarrow \text{NOT } (F0)$

**CPL F1 Complement Flag 1**

1 0 1 1	0 1 0 1
---------	---------

The setting of flag 1 is complemented; one is changed to zero, and zero is changed to one.

$(F1) \leftarrow \text{NOT } (F1)$

**DA A Decimal Adjust Accumulator**

0 1 0 1	0 1 1 1
---------	---------

The 8-bit accumulator value is adjusted to form two 4-bit Binary Coded Decimal (BCD) digits following the binary addition of BCD numbers. The carry bit C is affected. If the contents of bits 0-3 are greater than nine, or if AC is one, the accumulator is incremented by six.

The four high-order bits are then checked. If bits 4-7 exceed nine, or if C is one, these bits are increased by six. If an overflow occurs, C is set to one; otherwise, it is cleared to zero.

**Example:** Assume accumulator contains 10011011.  
 DA A ;ACC ADJUSTED TO 00000001  
 ;WITH C SET

C	AC	7	4	3	0	
0	0	1	0	0	1	1 0 1 1
						0 1 1 0
						ADD SIX TO BITS 0-5
0	0	1	0	1	0	0 0 0 1
						0 1 1 0
						ADD SIX TO BITS 4-7
1	0	0	0	0	0	0 0 0 1
						OVERFLOW TO C

**DEC A Decrement Accumulator**

0 0 0 0	0 1 1 1
---------	---------

The contents of the accumulator are decremented by one.

$(A) \leftarrow (A) - 1$

**Example:** Decrement contents of external data memory location 63.

```

MOV R0,#3FH      ;MOVE '3F' HEX TO REG 0
MOVX A,@R0       ;MOVE CONTENTS OF LOCATION 63
                 ;TO ACC
DEC A            ;DECREMENT ACC
MOVX @R0,A       ;MOVE CONTENTS OF ACC TO
                 ;LOCATION 63 IN EXPANDED
                 ;MEMORY

```

## **DEC R<sub>r</sub>   Decrement Register**

1 1 0 0	1 r r r
---------	---------

The contents of working register 'r' are decremented by one.

$(Rr) \leftarrow (Rr) - 1$                        $r = 0-7$

**Example:** DECR1: DEC R1                      ;DECREMENT CONTENTS OF REG 1

## **DIS I   Disable External Interrupt**

0 0 0 1	0 1 0 1
---------	---------

External interrupts are disabled. A low signal on the interrupt input pin has no effect.

## **DIS TCNTI   Disable Timer/Counter Interrupt**

0 0 1 1	0 1 0 1
---------	---------

Timer/counter interrupts are disabled. Any pending timer interrupt request is cleared. The interrupt sequence is not initiated by an overflow, but the timer flag is set and time accumulation continues.

## **DJNZ R<sub>r</sub>, address   Decrement Register and Test**

1 1 1 0	1 r r r	$a_7 a_6 a_5 a_4$	$a_3 a_2 a_1 a_0$
---------	---------	-------------------	-------------------

This is a 2-cycle instruction. Register 'r' is decremented and tested for zero. If the register contains all zeros, program control falls through to the next instruction. If the register contents are not zero, control jumps to the specified 'address'.

The address in this case must evaluate to 8-bits, that is, the jump must be to a location within the current 256-location page.

$(Rr) \leftarrow (Rr) - 1$                        $r = 0-7$   
 If Rr not 0  
 $(PC_{0-7}) \leftarrow \text{addr}$

Note: A 12-bit address specification does not cause an error if the DJNZ instruction and the jump target are on the same page. If the DJNZ instruction begins in location 255 of a page, it must jump to a target address on the following page.

**Example:** Increment values in data memory locations 50-54.

```

MOV R0,#50      ;MOVE '50' DEC TO ADDRESS
                 ;REG 0
MOV R3,#5       ;MOVE '5' DEC TO COUNTER
                 ;REG 3
INCRT: INC @R0   ;INCREMENT CONTENTS OF
                 ;LOCATION ADDRESSED BY
                 ;REG 0
INC R0          ;INCREMENT ADDRESS IN REG 0
DJNZ R3, INCRT  ;DECREMENT REG 3 — JUMP TO
                 ;'INCRT' IF REG 3 NONZERO
NEXT —         ;'NEXT' ROUTINE EXECUTED
                 ;IF R3 IS ZERO
  
```

## **EN I    Enable External Interrupt**

0 0 0 0	0 1 0 1
---------	---------

External interrupts are enabled. A low signal on the interrupt input pin initiates the interrupt sequence.

## **EN TCNTI    Enable Timer/Counter Interrupt**

0 0 1 0	0 1 0 1
---------	---------

Timer/counter interrupts are enabled. An overflow of this register initiates the interrupt sequence.

## **ENT0 CLK    Enable Clock Output**

0 1 1 1	0 1 0 1
---------	---------

The test 0 pin is enabled to act as the clock output. This function is disabled by a system reset.

**Example:** EMTST0: ENT0 CLK        ;ENABLE TO AS CLOCK OUTPUT

## **IN A,Pp    Input Port or Data to Accumulator**

0 0 0 0	1 0 p p
---------	---------

This is a 2-cycle instruction. Data present on port 'p' is transferred (read) to the accumulator.

(A) ← (Pp)                      p=1-2

**Example:** INP12: IN A,P1                   ;INPUT PORT 1 CONTENTS  
    ;TO ACC  
                   MOV R6,A               ;MOVE ACC CONTENTS TO  
    ;REG 6  
                   IN A,P2               ;INPUT PORT 2 CONTENTS  
    ;TO ACC  
                   MOV R7,A               ;MOVE ACC CONTENTS TO REG 7

## **INC A   Increment Accumulator**

---

0 0 0 1	0 1 1 1
---------	---------

The contents of the accumulator are incremented by one.

$(A) \leftarrow (A) + 1$

**Example:** Increment contents of location 100 in external data memory.

INCA: MOV R0,#100           ;MOVE '100' DEC TO ADDRESS  
    ;REG 0  
                   MOVX A,@R0       ;MOVE CONTENTS OF LOCATION  
    ;100 TO ACC  
                   INC A           ;INCREMENT A  
                   MOVX @R0,A   ;MOVE ACC CONTENTS TO  
    ;LOCATION 100

## **INC R<sub>r</sub>   Increment Register**

---

0 0 0 1	1 r r r
---------	---------

The contents of working register 'r' are incremented by one.

$(Rr) \leftarrow (Rr) + 1$                    r=0-7

**Example:** INCR0: INC R0                   ;INCREMENT ADDRESS REG 0

## **INC @R<sub>r</sub>   Increment Data Memory Location**

---

0 0 0 1	0 0 0 r
---------	---------

The contents of the resident data memory location addressed by register 'r' bits 0-5 are incremented by one.

$((Rr)) \leftarrow ((Rr)) + 1$                    r=0-1

**Example:** INCMD: MOV R1,#OFFH ;MOVE ONES TO REG 1  
                   INC @R1       ;INCREMENT LOCATION 63

## INS A,BUS    Strobed Input of BUS Data to Accumulator

0 0 0 0	1 0 0 0
---------	---------

This is a 2-cycle instruction. Data present on the BUS port is transferred (read) to the accumulator when the RD pulse is dropped. (Refer to section on programming memory expansion for details).

(A) ← (BUS)

**Example:**    INPBUS: INS A,BUS            ;INPUT BUS CONTENTS  
    ;TO ACC

## JBb address    Jump If Accumulator Bit is Set

b <sub>2</sub> b <sub>1</sub> b <sub>0</sub> 1	0 0 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
--	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if accumulator bit 'b' is set to one.

(PC<sub>0-7</sub>) ← addr                            If Bb=1  
 (PC) = (PC)+2                            If Bb=0

**Example:**    JB4IS1: JB4 NEXT            ;JUMP TO 'NEXT' ROUTINE  
    ;IF ACC BIT 4=1

## JC address    Jump If Carry Is Set

1 1 1 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the carry bit is set to one.

(PC<sub>0-7</sub>) ← addr                            If C=1  
 (PC) = (PC)+2                            If C=0

**Example:**    JC1: JC OVFLOW            ;JUMP TO 'OVFLOW' ROUTINE  
    ;IF C=1

## JF0 address    Jump If Flag 0 Is Set

1 0 1 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if flag 0 is set to one.

(PC<sub>0-7</sub>) ← addr                            If F0=1  
 (PC) = (PC)+2                            If F0=0

**Example:**    JF0IS1: JF0 TOTAL            ;JUMP TO 'TOTAL' ROUTINE  
    ;IF F0=1

## JF1 address    Jump If Flag 1 Is Set

0 1 1 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if flag 1 is set to one.

(PC<sub>0-7</sub>) ← addr                      If F1=1  
(PC) = (PC)+2                      IF F1=0

**Example:**    JF1IS1: JF1 FILBUF                      ;JUMP TO 'FILBUF'  
   ;ROUTINE IF F1=1

## JMP address    Direct Jump Within 2K Block

a <sub>10</sub> a <sub>9</sub> a <sub>8</sub> 1	0 1 0 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---	---------	---	---

This is a 2-cycle instruction. Bits 0-10 of the program counter are replaced with the directly-specified address. The setting of PC bit 11 is determined by the most recent SELECT MB instruction.

(PC<sub>8-10</sub>) ← addr 8-10  
(PC<sub>0-7</sub>) ← addr 0-7  
(PC<sub>11</sub>) ← DBF

**Example:**    JMP SUBTOT                      ;JUMP TO SUBROUTINE 'SUBTOT'  
                         JMP \$-6                      ;JUMP TO INSTRUCTION SIX LOCATIONS  
   ;BEFORE CURRENT LOCATION  
                         JMP 2FH                      ;JUMP TO ADDRESS '2F' HEX

## JMPP @A    Indirect Jump Within Page

1 0 1 1	0 0 1 1
---------	---------

This is a 2-cycle instruction. The contents of the program memory location pointed to by the accumulator are substituted for the 'page' portion of the program counter (PC bits 0-7).

(PC<sub>0-7</sub>) ← ((A))

**Example:**    Assume accumulator contains OFH.  
                 JMPPAG: JMPP @A                      ;JUMP TO ADDRESS STORED IN  
   ;LOCATION 15 IN CURRENT PAGE

## JNC address    Jump If Carry Is Not Set

1 1 1 0	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the carry bit is not set, that is, equals zero.

## INSTRUCTION SET

$(PC_{0-7}) \leftarrow \text{addr}$       If C=0  
 $(PC) = (PC)+2$       If C=1

**Example:** JC0: JNC NOVFO      ;JUMP TO 'NOVFO' ROUTINE  
                                 ;If C=0

### JNI address    Jump If Interrupt Input is Low

1 0 0 0	0 1 1 0	$a_7 a_6 a_5 a_4$	$a_3 a_2 a_1 a_0$
---------	---------	-------------------	-------------------

This is a 2-cycle instruction. Control passes to the specified address if the interrupt input signal is low (=0), that is, an external interrupt has been signaled. (This signal initiates an interrupt service sequence if the external interrupt is enabled.)

$(PC_{0-7}) \leftarrow \text{addr}$       If I=0  
 $(PC) = (PC)+2$       If I=1

**Example:** LOC 3: JNI EXTINT      ;JUMP TO 'EXTINT' ROUTINE  
                                 ;If I=0

### JNT0 address    Jump If Test 0 Is Low

0 0 1 0	0 1 1 0	$a_7 a_6 a_5 a_4$	$a_3 a_2 a_1 a_0$
---------	---------	-------------------	-------------------

This is a 2-cycle instruction. Control passes to the specified address, if the test 0 signal is low

$(PC_{0-7}) \leftarrow \text{addr}$       If T0=0  
 $(PC) = (PC)+2$       If T0=1

**Example:** JTOLOW: JNT0 60      ;JUMP TO LOCATION 60 DEC  
                                 ;If T0=0

### JNT1 address    Jump If Test 0 Is High

0 1 0 0	0 1 1 0	$a_7 a_6 a_5 a_4$	$a_3 a_2 a_1 a_0$
---------	---------	-------------------	-------------------

This is a 2-cycle instruction. Control passes to the specified address, if the test 0 signal is high.

$(PC_{0-7}) \leftarrow \text{addr}$       If T0=1  
 $(PC) = (PC)+2$       If T0=0

### JNZ address    Jump If Accumulator Is Not Zero

1 0 0 1	0 1 1 0	$a_7 a_6 a_5 a_4$	$a_3 a_2 a_1 a_0$
---------	---------	-------------------	-------------------

This is a 2-cycle instruction. Control passes to the specified address if the accumulator contents are nonzero at the time this instruction is executed.

$(PC_{0-7}) \leftarrow \text{addr}$       If A $\neq$ 0  
 $(PC) = (PC)+2$       If A=0

**Example:** JACCN0: JNZ 0ABH      ;JUMP TO LOCATION 'AB' HEX  
                                 ;IF ACC VALUE IS NONZERO



## JTF address    Jump If Timer Flag Is Set

0 0 0 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the timer flag is set to one, that is, the timer/counter register has overflowed. Testing the timer flag resets it to zero. (This overflow initiates an interrupt service sequence if the timer-overflow interrupt is enabled.)

(PC<sub>0-7</sub>) ← addr                      If TF=1  
(PC) = (PC)+2                      If TF=0

**Example:**    JTF1: JTF TIMER                      ;JUMP TO 'TIMER' ROUTINE  
   ;IF TF=1

## JT0 address    Jump If Test 0 Is High

0 0 1 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the test 0 signal is high (=1).

(PC<sub>0-7</sub>) ← addr                      If T0=1  
(PC) = (PC)+2                      If T0=0

**Example:**    JT0HI: JT0 53                      ;JUMP TO LOCATION 53 DEC  
   ;IF T0=1

## JT1 address    Jump If Test 1 Is High

0 1 0 1	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the test 1 signal is high (=1).

(PC<sub>0-7</sub>) ← addr                      If T1=1  
(PC) = (PC)+2                      If T1=0

**Example:**    JT1HI: JT1 COUNT                      ;JUMP TO 'COUNT' ROUTINE  
   ;IF T1=1

## JZ address    Jump If Accumulator Is Zero

1 1 0 0	0 1 1 0	a <sub>7</sub> a <sub>6</sub> a <sub>5</sub> a <sub>4</sub>	a <sub>3</sub> a <sub>2</sub> a <sub>1</sub> a <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Control passes to the specified address if the accumulator contains all zeros at the time this instruction is executed.

(PC<sub>0-7</sub>) ← addr                      If A=0  
(PC) = (PC)+2                      If A≠0

**Example:**    JACCO: JZ OA3H                      ;JUMP TO LOCATION 'A3' HEX  
   ;IF ACC VALUE IS ZERO

0 0 1 0	0 0 1 1
---------	---------

d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---	---

This is a 2-cycle instruction. The 8-bit value specified by 'data' is loaded in the accumulator.

(A) ← data

**Example:** MOV A,#0A3H ;MOVE 'A3' HEX TO ACC

1 1 0 0	0 1 1 1
---------	---------

The contents of the program status word are moved to the accumulator.

(A) ← (PSW)

**Example:** Jump to 'RB1SET' routine if PSW bank switch, bit 4, is set.

```
BSCHK: MOV A,PSW      ;MOVE PSW CONTENTS TO ACC
        JB4 RB1SET    ;JUMP TO 'RB1SET' IF ACC
                        ;BIT 4=1
```

1 1 1 1	1 r r r
---------	---------

8-bits of data are moved from working register 'r' into the accumulator.

$$(A) \leftarrow (Rr) \quad r=0-7$$

**Example:** MAR: MOV A,R3                   ;MOVE CONTENTS OF REG 3  
  ;TO ACC

1 1 1 1	0 0 0 r
---------	---------

The contents of the resident data memory location addressed by bits 0-5 of register 'r' are moved to the accumulator. Register 'r' contents are unaffected.

$$(A) \leftarrow ((Rr)) \quad r=0-1$$

**Example:** Assume R1 contains 01110110.  
MADM: MOV A,@R1 ;MOVE CONTENTS OF DATA MEM  
;LOCATION 54 TO ACC

**MOV A,T Move Timer/Counter Contents to Accumulator**

0 1 0 0	0 0 1 0
---------	---------

The contents of the timer/event-counter register are moved to the accumulator.

$(A) \leftarrow (T)$

**Example:** Jump to "EXIT" routine when timer reaches '64', that is, when bit 6 set — assuming initialization 64,

```

TIMCHK: MOV A,T      ;MOVE TIMER CONTENTS TO
                  ;ACC
          JB6 EXIT    ;JUMP TO 'EXIT' IF ACC BIT
                  ;6=1

```

**MOV PSW,A Move Accumulator Contents to PSW**

1 1 0 1	0 1 1 1
---------	---------

The contents of the accumulator are moved into the program status word. All condition bits and the stack pointer are affected by this move.

$(PSW) \leftarrow (A)$

**Example:** Move up stack pointer by two memory locations, that is, increment the pointer by one.

```

INCPTR: MOV A,PSW    ;MOVE PSW CONTENTS TO ACC
          INC A        ;INCREMENT ACC BY ONE
          MOV PSW,A    ;MOVE ACC CONTENTS TO PSW

```

**MOV R<sub>r</sub>,A Move Accumulator Contents to Register**

1 0 1 0	1 r r r
---------	---------

The contents of the accumulator are moved to register 'r'.

$(R_r) \leftarrow (A)$   $r=0-7$

**Example:** MRA: MOV R0,A ;MOVE CONTENTS OF ACC TO  
;REG 0

**MOV R<sub>r</sub>,#data Move Immediate Data to Register**

1 0 1 1	1 r <sub>2</sub> r <sub>1</sub> r <sub>0</sub>	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	--	---	---

This is a 2-cycle instruction. The 8-bit value specified by 'data' is moved to register 'r'.

$(R_r) \leftarrow \text{data}$   $r=0-7$

**Examples:** MIR4: MOV R4,#HEXTEN ;THE VALUE OF THE SYMBOL  
;‘HEXTEN’ IS MOVED INTO  
;REG 4  
MIR 5: MOV R5,#PI\*(R\*R);THE VALUE OF THE  
;EXPRESSION ‘PI\*(R\*R)  
;IS MOVED INTO REG 5  
MIR 6: MOV R6, #0ADH ;‘AD’ HEX IS MOVED INTO  
;REG 6

### MOV @R<sub>r</sub>,A    Move Accumulator Contents to Data Memory

1 0 1 0	0 0 0 r
---------	---------

The contents of the accumulator are moved to the resident data memory location whose address is specified by bits 0-5 of register 'r'. Register 'r' contents are unaffected.

$$((Rr)) \leftarrow (A) \quad r=0-1$$

**Example:** Assume R0 contains 11000111.  
MDMA: MOV @R0,A ;MOVE CONTENTS OF ACC TO  
;LOCATION 7 (REG 7)

### MOV @R<sub>r</sub>,#data    Move Immediate Data to Data Memory

1 0 1 1	0 0 0 r
---------	---------

d7 d6 d5 d4	d3 d2 d1 d0
-------------	-------------

This is a 2-cycle instruction. The 8-bit value specified by 'data' is moved to the resident data memory location addressed by register 'r', bits 0-5.

```
((Rr)) ← data      r=0-1
```

**Examples:** Move the hexadecimal value AC3F to locations 62-63.

```
MIDM: MOV R0,#62      ;MOVE '62' DEC TO ADDR REG 0
      MOV @R0,#0ACH    ;MOVE 'AC' HEX TO LOCATION 62
      INC R0           ;INCREMENT REG 0 TO '63'
      MOV @R0,#3FH     ;MOVE '3F' HEX TO LOCATION 63
```

## MOV T,A    Move Accumulator Contents to Timer/Counter

0 1 1 0	0 0 1 0
---------	---------

The contents of the accumulator are moved to the timer/event-counter register.

$$(T) \leftarrow (A)$$

**Example:** Initialize and start event counter.

INITEC:	CLR A	;CLEAR ACC TO ZEROS
	MOV T,A	;MOVE ZEROS TO EVENT COUNTER
	STRT CNT	;START COUNTER

**MOVD A,Pp Move Port 4-7 Data to Accumulator**

0 0 0 0	1 1 p p
---------	---------

This is a 2-cycle instruction. Data on 8243 port 'p' is moved (read) to accumulator bits 0-3. Accumulator bits 4-7 are zeroed.

(0-3) ← (Pp)                      p=4-7  
(4-7) ← 0

Note: Bits 0-1 of the opcode are used to represent ports 4-7. If you are coding in binary rather than assembly language, the mapping is as follows:

Bits	1	0	Port
0	0		4
0	1		5
1	0		6
1	1		7

**Example:** INPPT5: MOVD A,P5                      ;MOVE PORT 5 DATA TO ACC  
   ;BITS 0-3, ZERO ACC BITS 4-7

**MOVD Pp,A Move Accumulator Data to Port 4-7**

0 0 1 1	1 1 p p
---------	---------

Data in accumulator bits 0-3 is moved (written) to 8243 port 'p'. Accumulator bits 4-7 are unaffected. (See NOTE above regarding port mapping.)

(Pp) ← (A<sub>0-3</sub>)                      p=4-7

**Example:** Move data in accumulator to ports 4 and 5.  
OUTP45: MOVD P4,A                      ;MOVE ACC BITS 0-3 TO PORT 4  
                         SWAP A                      ;EXCHANGE ACC BITS 0-3 AND 4-7  
                         MOVD P5,A                      ;MOVE ACC BITS 0-3 TO PORT 5

**MOVP A,@A Move Current Page Data to Accumulator**

1 0 1 0	0 0 1 1
---------	---------

The contents of the program memory location addressed by the accumulator are moved to the accumulator. Only bits 0-7 of the program counter are affected, limiting the program memory reference to the current page. The program counter is restored following this operation

(PC<sub>0-7</sub>) ← (A)  
(A) ← ((PC))

Note: This is a 1-byte, 2-cycle instruction. If it appears in location 255 of a program memory page, @A addresses a location in the following page.

```

Example:  MOV128: MOV A,#128      ;MOVE '128' DEC TO ACC
                MOVP, A,@A          ;CONTENTS OF 129th LOCATION
                                     ;IN CURRENT PAGE ARE MOVED TO
                                     ;ACC

```

## MOV P3 A,@A    Move Page 3 Data to Accumulator

1 1 1 0	0 0 1 1
---------	---------

This is a 2-cycle instruction. The contents of the program memory location (within page 3) addressed by the accumulator are moved to the accumulator. The program counter is restored following this operation.

$$\begin{aligned} (PC_{0-7}) &\leftarrow (A) \\ (PC_{8-10}) &\leftarrow 011 \\ (A) &\leftarrow ((PC)) \end{aligned}$$

**Example:** Look up ASCII equivalent of hexadecimal code in table contained at the beginning of page 3. Note that ASCII characters are designated by a 7-bit code; the eighth bit is always reset.

```
TABSCH: MOV A,#0B8H ;MOVE 'B8' HEX TO ACC (10111000)
        ANL A,#7FH  ;LOGICAL AND ACC TO MASK BIT
                        ;7 (00111000)
        MOVP3, A,@A ;MOVE CONTENTS OF LOCATION
                        ;'38' HEX IN PAGE 3 TO ACC
                        ;(ASCII '8')
```

Access contents of location in page 3 labelled TAB1.  
Assume current program location is not in page 3.

```
TABSCH: MOV A,#LOW TAB1 ;ISOLATE BITS 0-7 OF LABEL
          ;ADDRESS VALUE
          MOVP3 A,@A      ;MOVE CONTENTS OF PAGE 3
          ;LOCATION LABELED 'TAB1'
          ;TO ACC
```

### **MOVX A,@R<sub>r</sub>    Move External-Data-Memory Contents to Accumulator**

1000	000r
------	------

This is a 2-cycle instruction. The contents of the external data memory location addressed by register 'r' are moved to the accumulator. Register 'r' contents are unaffected.

$$(A) \leftarrow ((Rr)) \quad r=0-1$$

**Example:** Assume R1 contains 01110110.

```
MAXDM: MOVX A,@R1      ;MOVE CONTENTS OF LOCATION
                        ;118 TO ACC
```

**MOVX @R<sub>r</sub>,A    Move Accumulator Contents to External Data Memory**

1	0	0	1	0	0	0	r
---	---	---	---	---	---	---	---

This is a 2-cycle instruction. The contents of the accumulator are moved to the external data memory location addressed by register 'r'. Register 'r' contents are unaffected.

$$((Rr)) \leftarrow A$$

**Example:** Assume R0 contains 11000111.

```
MXDMA: MOVX @R0,A      ;MOVE CONTENTS OF ACC TO
                        ;LOCATION 199 IN EXPANDED
                        ;DATA MEMORY
```

**NOP    The NOP Instruction**

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

No operation is performed. Execution continues with the following instruction.

**ORL A,R<sub>r</sub>    Logical OR Accumulator With Register Mask**

0	1	0	0	1	r	r	r
---	---	---	---	---	---	---	---

Data in the accumulator is logically ORed with the mask contained in working register 'r'.

$$(A) \leftarrow (A) \text{ OR } (Rr) \quad r=0-7$$

**Example:** ORREG: ORL A,R4                    ;'OR' ACC CONTENTS WITH  
   ;MASK IN REG 4

**ORL A,@R<sub>r</sub>    Logical OR Accumulator With Memory Mask**

0	1	0	0	0	0	0	r
---	---	---	---	---	---	---	---

Data in the accumulator is logically ORed with the mask contained in the resident data memory location referenced by register 'r', bits 0-5.

$$(A) \leftarrow (A) \text{ OR } ((Rr)) \quad r=0-1$$

**Example:** ORDM: MOV R0,#3FH                ;MOVE '3F' HEX TO REG 0  
   ;'OR' ACC CONTENTS WITH MASK  
   ;IN LOCATION 63

**ORL A,#data    Logical OR Accumulator With Immediate Mask**

0	1	0	0	0	0	1	1	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
---	---	---	---	---	---	---	---	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

This is a 2-cycle instruction. Data in the accumulator is logically ORed with an immediately-specified mask.

$$(A) \leftarrow (A) \text{ OR data}$$

**Example:** ORID: ORL A,#'X'                    ;'OR' ACC CONTENTS WITH MASK  
   ;01011000 (ASCII VALUE OF 'X')

**ORL BUS,#data Logical OR BUS With Immediate Mask**

1 0 0 0	1 0 0 0	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data on the BUS port is logically ORed with an immediately-specified mask. This instruction assumes prior specification of an 'OUTL BUS,A' instruction.

(BUS) ← (BUS) OR data

**Example:** ORBUS: ORL BUS,#HEXMSK ;'OR' BUS CONTENTS WITH  
;MASK EQUAL VALUE OF SYMBOL  
; 'HEXMSK'

**ORL Pp, #data Logical OR Port 1 or 2 With Immediate Mask**

1 0 0 0	1 0 p p	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data on port 'p' is logically ORed with an immediately-specified mask.

(Pp) ← (Pp) OR data                      p=1-2

**Example:** ORP1: ORL P1, #0FFH ;'OR' PORT 1 CONTENTS WITH  
;MASK 'FF' HEX ( SET PORT 1  
;TO ALL ONES)

**ORLD Pp,A Logical OR Port 4-7 With Accumulator Mask**

1 0 0 0	1 1 p p
---------	---------

Data on port 'p' is logically ORed with the digit mask contained in accumulator bits 0-3.

(Pp) ← (Pp) OR (A<sub>0-3</sub>)                      p=4-7

**Example:** ORP7: ORLD P7,A ;'OR' PORT 7 CONTENTS  
;WITH ACC BITS 0-3

**OUTL BUS,A Output Accumulator Data to BUS**

0 0 0 0	0 0 1 0
---------	---------

Data residing in the accumulator is transferred (written) to the BUS port and latched. The latched data remains valid until altered by another OUTL instruction. Any other instruction requiring use of the BUS port (except INS) destroys the contents of the BUS latch. This includes expanded memory operations (such as the MOVX instruction). Logical operations on BUS data (AND, OR) assume the OUTL BUS,A instruction has been issued previously.

(BUS) ← (A)

**Example:** OUTLBP: OUTL BUS,A ;OUTPUT ACC CONTENTS TO BUS



**OUTL Pp,A    Output Accumulator Data to Port 1 or 2**

---

0 0 1 1	1 0 p p
---------	---------

Data residing in the accumulator is transferred (written) to port 'p' and latched.

$$(Pp) \leftarrow (A)$$
$$p=1-2$$

**Example:**    OUTLP: MOV A,R7                    ;MOVE REG 7 CONTENTS TO ACC  
                 OUTL P2,A                    ;OUTPUT ACC CONTENTS TO PORT 2  
                 MOV A,R6                    ;MOVE REG 6 CONTENTS TO ACC  
                 OUTL P1,A                    ;OUTPUT ACC CONTENTS TO PORT 1

**RET    Return Without PSW Restore**

---

1 0 0 0	0 0 1 1
---------	---------

This is a 2-cycle instruction. The stack pointer (PSW bits 0-2) is decremented. The program counter is then restored from the stack. PSW bits 4-7 are not restored.

$$(SP) \leftarrow (SP)-1$$
$$(PC) \leftarrow ((SP))$$
**RETR    Return With PSW Restore**

---

1 0 0 1	0 0 1 1
---------	---------

This is a 2-cycle instruction. The stack pointer is decremented. The program counter and bits 4-7 of the PSW are then restored from the stack. Note that RETR should be used to return from an interrupt, but should not be used within the interrupt service routine as it signals the end of an interrupt routine.

$$(SP) \leftarrow (SP)-1$$
$$(PC) \leftarrow ((SP))$$
$$(PSW\ 4-7) \leftarrow ((SP))$$
**RL A    Rotate Left Without Carry**

---

1 1 1 0	0 1 1 1
---------	---------

The contents of the accumulator are rotated left one bit. Bit 7 is rotated into the bit 0 position.

$$(AN+1) \leftarrow (An)$$
$$(A0) \leftarrow (A7)$$
$$n=0-6$$

**Example:**    Assume accumulator contains 10110001.  
                 RLNC: RL A                    ;NEW ACC CONTENTS ARE 01100011.

**RLC A Rotate Left Through Carry**

---

1	1	1	1
0	1	1	1

The contents of the accumulator are rotated left one bit. Bit 7 replaces the carry bit; the carry bit is rotated into the bit 0 position.

$$(AN+1) \leftarrow (An)$$
$$n=0-6$$
$$(A0) \leftarrow (C)$$
$$(C) \leftarrow (A7)$$

**Example:** Assume accumulator contains a 'signed' number; isolate sign without changing value.

RLTC: CLR C

;CLEAR CARRY TO ZERO

RLC A

;ROTATE ACC LEFT, SIGN

;BIT (7) IS PLACED IN CARRY

RR A

;ROTATE ACC RIGHT — VALUE

(BITS 0-6) IS RESTORED,

;CARRY UNCHANGED, BIT 7

;IS ZERO

**RR A Rotate Right Without Carry**

---

0	1	1	1
0	1	1	1

The contents of the accumulator are rotated right one bit. Bit 0 is rotated into the bit 7 position

$$(An) \leftarrow (AN+1)$$
$$n=0-6$$
$$(A7) \leftarrow (A0)$$

**Example:** Assume accumulator contains 10110001.

RRNC: RR A

;NEW ACC CONTENTS ARE 11011000

**RRC A Rotate Right Through Carry**

---

0	1	1	0
0	1	1	1

The contents of the accumulator are rotated right one bit. Bit 0 replaces the carry bit; the carry bit is rotated into the bit 7 position.

$$(An) \leftarrow (An+1)$$
$$n=0-6$$
$$(A7) \leftarrow (C)$$
$$(C) \leftarrow (A0)$$

**Example:** Assume carry is not set and accumulator contains 10110001.

RRTC: RRC A

;CARRY IS SET AND ACC

;CONTAINS 01011000

**SEL MBO    Select Memory Bank 0**

---

1	1	1	0
0	1	0	1

PC bit 11 is set to zero on next branch instruction.  
All references to program memory addresses fall within  
the range 0-2047.

(DBF) ← 0

**Example:** Assume program counter contains 834 Hex and the  
carry bit is set.

SEL MBO	;SELECT MEMORY BANK 0
JC \$+20	;IF C=1, JUMP TO LOCATION
	;48 HEX

**SEL MB1    Select Memory Bank 1**

---

1	1	1	1
0	1	0	1

PC bit 11 is set to one on next branch instruction.  
All references to program memory addresses fall  
within the range 2048-4095.

(DBF) ← 1

**SEL RB0    Select Register Bank 0**

---

1	1	0	0
0	1	0	1

PSW bit 4 is set to zero. References to working  
registers 0-7 address data memory locations 0-7.  
This is the recommended setting for normal program  
execution.

(BS) ← 0

**SEL RB1    Select Register Bank 1**

---

1	1	0	1
0	1	0	1

PSW bit 4 is set to one. References to working registers  
0-7 address data memory locations 24-31. This is the  
recommended setting for interrupt service routines,  
since locations 0-7 are left intact. The setting of  
PSW bit 4 in effect at the time of an interrupt is  
restored by the RETR instruction when the interrupt  
service routine is completed.

(BS) ← 1

**Example:** Assume an external interrupt has occurred, control  
has passed to program memory location 3, and PSW bit  
4 was zero before the interrupt.

LOC3: JN1 INIT	;JUMP TO ROUTINE 'INIT' IF
	;INTERRUPT INPUT IS ZERO

## INSTRUCTION SET

---

```
INIT: MOV R7,A      ;MOVE ACC CONTENTS TO
                   ;LOCATION 7
      SEL RB1       ;SELECT REG BANK 1
      MOV R7,#0FAH  ;MOVE 'FA' HEX TO LOCATION 31
      .
      .
      .
      SEL RB0       ;SELECT REG BANK 0
      MOV A,R7      ;RESTORE ACC FROM LOCATION 7
      RETR          ;RETURN — RESTORE PC AND PSW
```

### **STOP TCNT** Stop Timer/Event-Counter

---

0	1	1	0	0	1	0	1
---	---	---	---	---	---	---	---

This instruction is used to stop both time accumulation and event counting.

**Example:** Disable interrupt, but jump to interrupt routine after eight overflows and stop timer. Count overflows in register 7.

```
START: DIS TCNTI    ;DISABLE TIMER INTERRUPT
      CLR A         ;CLEAR ACC TO ZEROS
      MOV T,A       ;MOVE ZEROS TO TIMER
      MOV R7,A      ;MOVE ZEROS TO REG 7
      STRT T        ;START TIMER
MAIN:  JTF COUNT     ;JUMP TO ROUTINE 'COUNT'
                   ;IF TF=1 AND CLEAR TIMER FLAG
      JMP MAIN      ;CLOSE LOOP
COUNT: INC R7       ;INCREMENT REG 7
      MOV A,R7      ;MOVE REG 7 CONTENTS TO ACC
      JB3 INT       ;JUMP TO ROUTINE 'INT' IF ACC
                   ;BIT 3 IS SET (REG 7=8)
      JMP MAIN      ;OTHERWISE RETURN TO ROUTINE
                   ;MAIN
      .
      .
      .
INT:   STOP TCNT     ;STOP TIMER
      JMP 7H         ;JUMP TO LOCATION 7 (TIMER)
                   ;INTERRUPT ROUTINE
```

**STRT CNT Start Event Counter**

---

0 1 0 0	0 1 0 1
---------	---------

The test 1 (T1) pin is enabled as the event-counter input and the counter is started. The event-counter register is incremented with each high-to-low transition on the T1 pin.

**Example:** Initialize and start event counter. Assume overflow is desired with first T1 input.

```
STARTC: EN TCNTI      ;ENABLE COUNTER INTERRUPT
          MOV A,#0FFH  ;MOVE 'FF' HEX (ONES) TO
                      ;ACC
          MOV T,A       ;MOVE ONES TO COUNTER
          STRT CNT      ;ENABLE TIAS COUNTER
                      ;INPUT AND START
```

**STRT T Start Timer**

---

0 1 0 1	0 1 0 1
---------	---------

Timer accumulation is initiated in the timer register. The register is incremented every 32 instruction cycles. The prescaler which counts the 32 cycles is cleared but the timer register is not.

**Example:** Initialize and start timer.

```
STARTT: CLR A          ;CLEAR ACC TO ZEROS
          MOV T,A       ;MOVE ZEROS TO TIMER
          EN TCNTI      ;ENABLE TIMER INTERRUPT
          STRT T        ;START TIMER
```

**SWAP A Swap Nibbles Within Accumulator**

---

0 1 0 0	0 1 1 1
---------	---------

Bits 0-3 of the accumulator are swapped with bits 4-7 of the accumulator.

$(A_{4-7}) \rightleftarrows (A_{0-3})$

**Example:** Pack bits 0-3 of locations 50-51 into location 50.

```
PCKDIG: MOV R0, #50    ;MOVE '50' DEC TO REG 0
          MOV R1, #51    ;MOVE '51' DEC TO REG 1
          XCHD A,@R0     ;EXCHANGE BITS 0-3 OF ACC
                      ;AND LOCATION 50
          SWAP A         ;SWAP BITS 0-3 AND 4-7 OF ACC
          XCHD A,@R1     ;EXCHANGE BITS 0-3 OF ACC AND
                      ;LOCATION 51
          MOV @R0,A      ;MOVE CONTENTS OF ACC TO
                      ;LOCATION 50
```

**XCH A,R<sub>r</sub>   Exchange Accumulator-Register Contents**

---

0 0 1 0	1 r r r
---------	---------

The contents of the accumulator and the contents of working register 'r' are exchanged.

(A)  $\longleftrightarrow$  (R<sub>r</sub>)                      r=0-7

**Example:** Move PSW contents to Reg 7 without losing accumulator contents.

```
XCHAR7: XCH A,R7      ;EXCHANGE CONTENTS OF REG 7
          ;AND ACC
          MOV A, PSW    ;MOVE PSW CONTENTS TO ACC
          XCH A,R7      ;EXCHANGE CONTENTS OF REG 7
          ;AND ACC AGAIN
```

**XCH A,@R<sub>r</sub>   Exchange Accumulator and Data Memory Contents**

---

0 0 1 0	0 0 0 r
---------	---------

The contents of the accumulator and the contents of the resident data memory location addressed by bits 0-5 of register 'r' are exchanged. Register 'r' contents are unaffected.

(A)  $\longleftrightarrow$  — ((R<sub>r</sub>))                      r=0-1

**Example:** Decrement contents of location 52.

```
DEC52: MOV R0,#52      ;MOVE '52' DEC TO ADDRESS
          ;REG 0
          XCH A,@R0     ;EXCHANGE CONTENTS OF ACC
          ;AND LOCATION 52
          DEC A          ;DECREMENT ACC CONTENTS
          XCH A,@R0     ;EXCHANGE CONTENTS OF ACC
          ;AND LOCATION 52 AGAIN
```

**XCHD A,@R<sub>r</sub>   Exchange Accumulator and Data Memory 4-Bit Data**

---

0 0 1 1	0 0 0 r
---------	---------

This instruction exchanges bits 0-3 of the accumulator with bits 0-3 of the data memory location addressed by bits 0-5 of register 'r'. Bits 4-7 of the accumulator, bits 4-7 of the data memory location, and the contents of register 'r' are unaffected.

(A<sub>0-3</sub>)  $\longleftrightarrow$  ((R<sub>r</sub>0-3))                      r=0-1

**Example:** Assume program counter contents have been stacked in locations 22-23.

```
XCHNIB: MOV R0,#23    ;MOVE '23' DEC TO REG 0
          CLR A        ;CLEAR ACC TO ZEROS
          XCHD A,@R0   ;EXCHANGE BITS 0-3 OF ACC
                      ;AND LOCATION 23 (BITS 8-11
                      ;OF PC ARE ZEROED, ADDRESS
                      ;REFERS TO PAGE 0)
```

## **XRL A,R<sub>r</sub> Logical XOR Accumulator With Register Mask**

1 1 0 1	1 r r r
---------	---------

Data in the accumulator in EXCLUSIVE ORed with the mask contained in working register 'r'.

$(A) \leftarrow (A) \text{ XOR } (R_r) \quad r=0-7$

**Example:** XORREG: XRL A,R5 ;'XOR' ACC CONTENTS WITH  
;MASK IN REG 5

## **XRL A,@R<sub>r</sub> Logical XOR Accumulator With Memory Mask**

1 1 0 1	0 0 0 r
---------	---------

Data in the accumulator is EXCLUSIVE ORed with the mask contained in the data memory location addressed by register 'r', bits 0-5.

$(A) \leftarrow (A) \text{ XOR } ((R_r)) \quad r=0-1$

**Example:** XORDM: MOV R1, #20H ;MOVE '20' HEX TO REG 1  
XRL A,@R1 ;'XOR' ACC CONTENTS WITH MASK  
;IN LOCATION 32

## **XRL A,#data Logical XOR Accumulator With Immediate Mask**

1 1 0 1	0 0 1 1	d <sub>7</sub> d <sub>6</sub> d <sub>5</sub> d <sub>4</sub>	d <sub>3</sub> d <sub>2</sub> d <sub>1</sub> d <sub>0</sub>
---------	---------	---	---

This is a 2-cycle instruction. Data in the accumulator is EXCLUSIVE ORed with an immediately-specified mask.

$(A) \leftarrow (A) \text{ XOR } \text{data}$

**Example:** XORID: XOR A,#HEXTEN ;XOR CONTENTS OF ACC WITH  
;MASK EQUAL VALUE OF SYMBOL  
;'HEXTEN'

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active

Example: Address: 12345 Main Street, New York, NY 10001  
Phone: 212-555-1234  
Fax: 212-555-5678  
E-mail: john.doe@example.com  
Web: www.example.com  
Company: Example Corp.  
Department: Sales  
Position: Sales Representative  
Start Date: 01/01/2000  
End Date: 12/31/2000  
Status: Active



## Chapter 5

# APPLICATION EXAMPLES



# APPLICATION EXAMPLES

5.0 Introduction .....	5-1
5.1 Hardware Examples .....	5-1
5.2 Software Examples .....	5-13

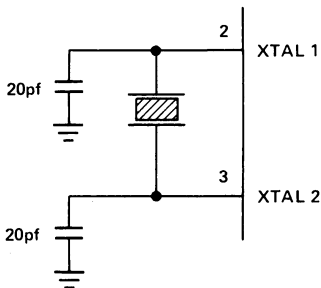
# APPLICATION EXAMPLES

## 5.0 Introduction

The following chapter is organized in two sections, Hardware and Software. The hardware section gives examples of some typical configurations of MCS-48 components while software section gives assembly language listings of some common applications routines.

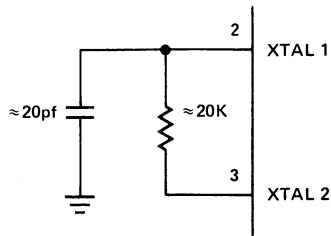
## 5.1 Hardware Examples

---



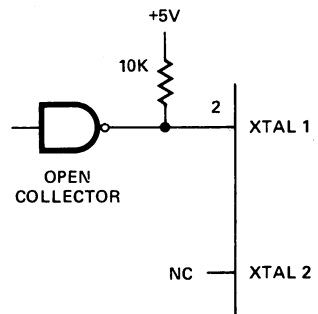
1.6 MHz

CRYSTAL



$\approx 3\text{ MHz}$

RC

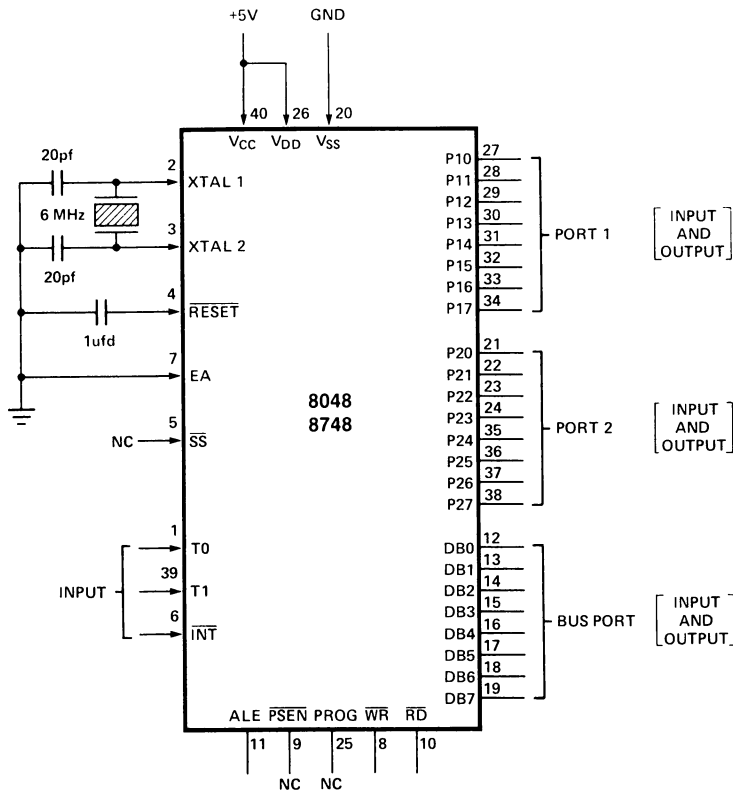


1.6 MHz

EXTERNAL

---

## FREQUENCY REFERENCE OPTIONS

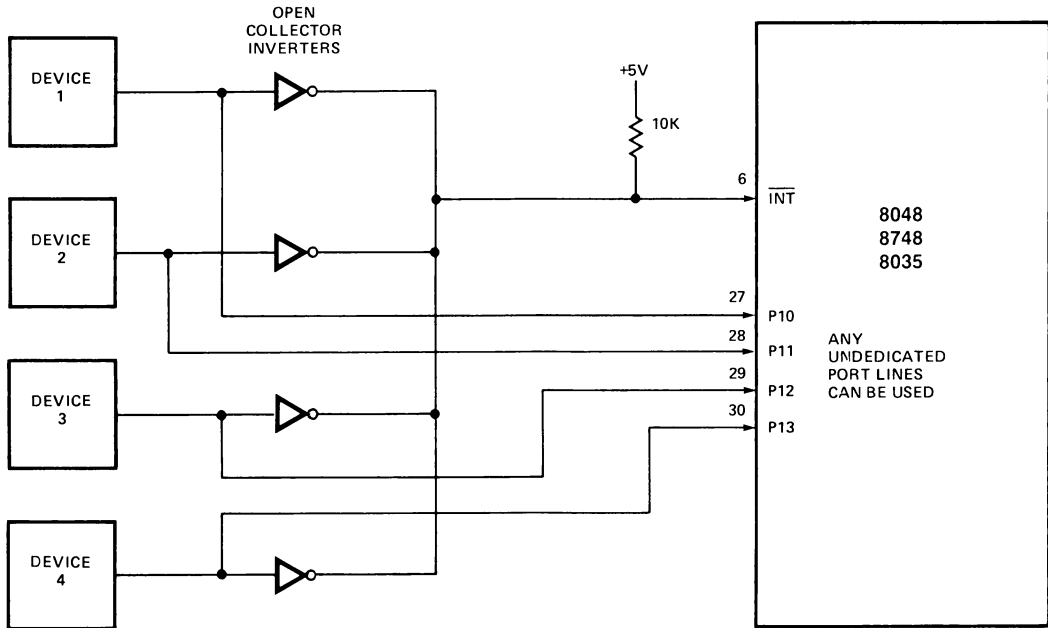


- All inputs and outputs standard TTL compatible
- P1 and P2 outputs drive 5V CMOS directly others require 10-50K pullup

**XTAL: Series Resonant  
AT Cut  
1 to 6 MHz**

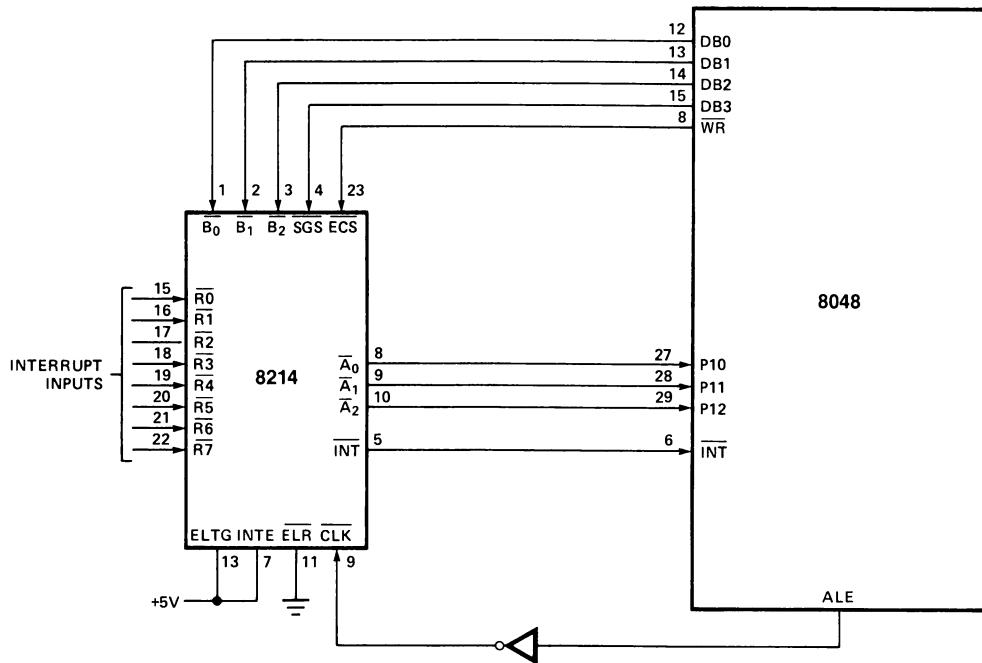
**THE STAND ALONE 8048**

## APPLICATION EXAMPLES



- All devices equal priority
- Processor polls Port 1 to determine interrupting device

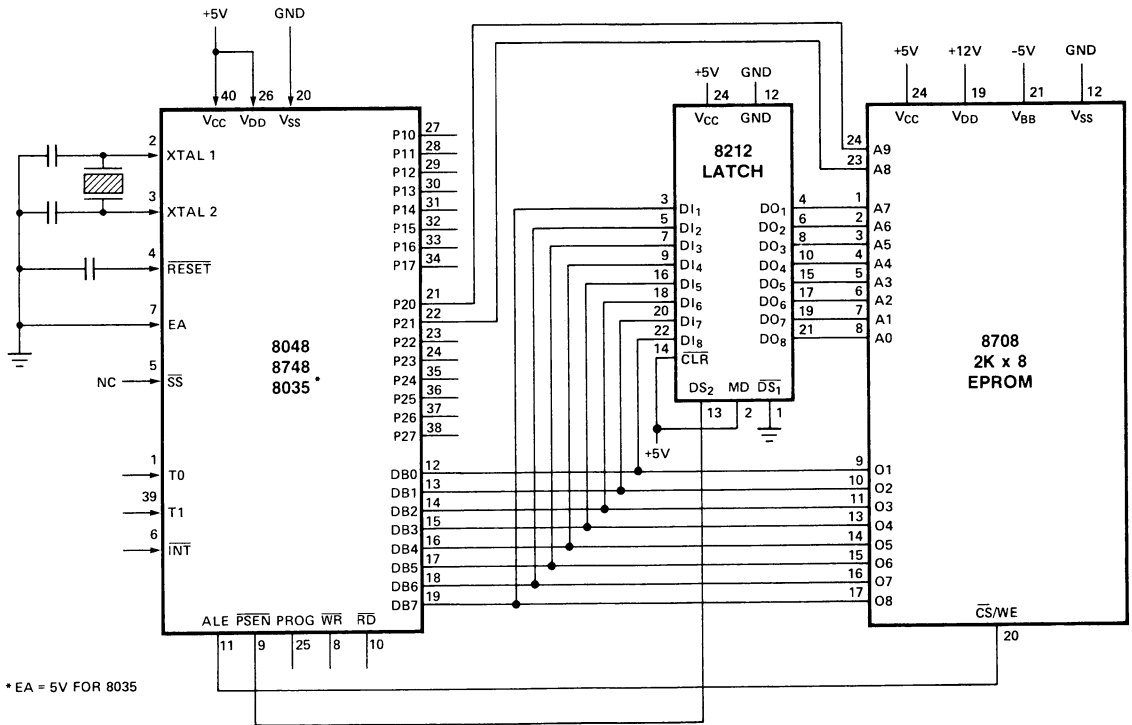
## MULTIPLE INTERRUPT SOURCES



- Processor polls Port 1 to determine interrupting device
- Processor sets priority level by writing 4-bits to 8214

## MULTIPLE INTERRUPTS WITH PRIORITY LEVELS

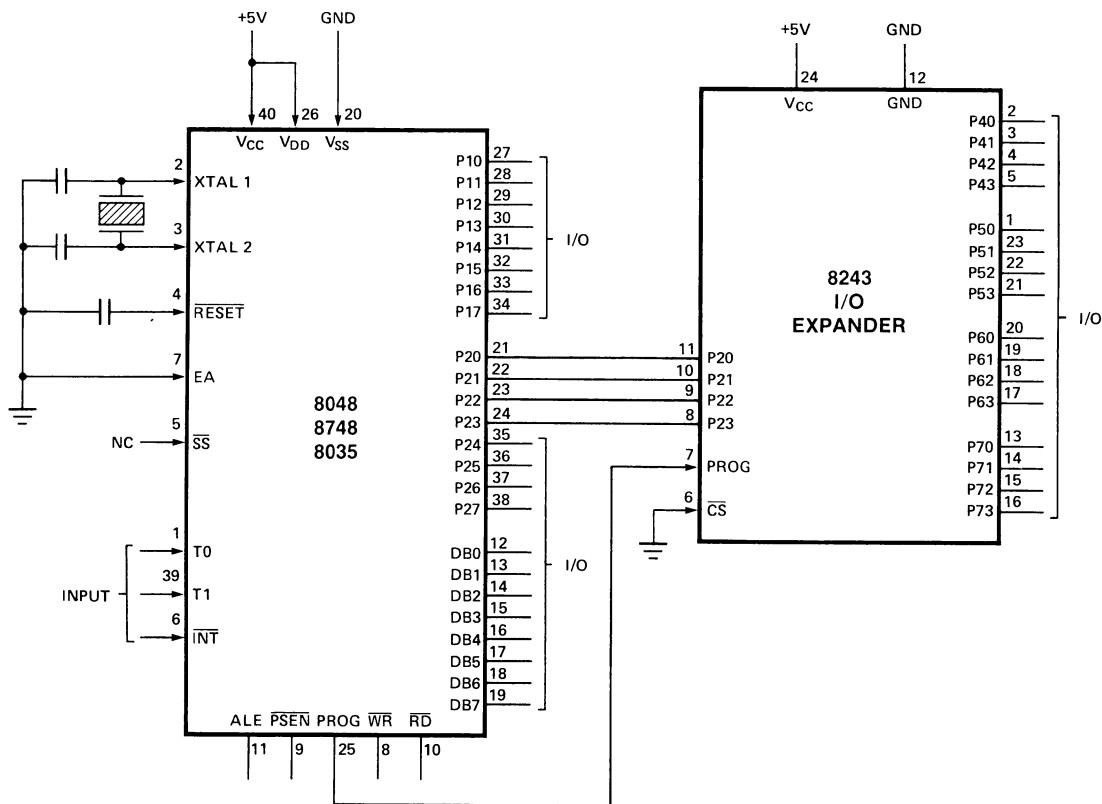
## APPLICATION EXAMPLES



- 8212 serves as address latch
- Address is valid while ALE is high and is latched when ALE goes low

## EXTERNAL PROGRAM MEMORY

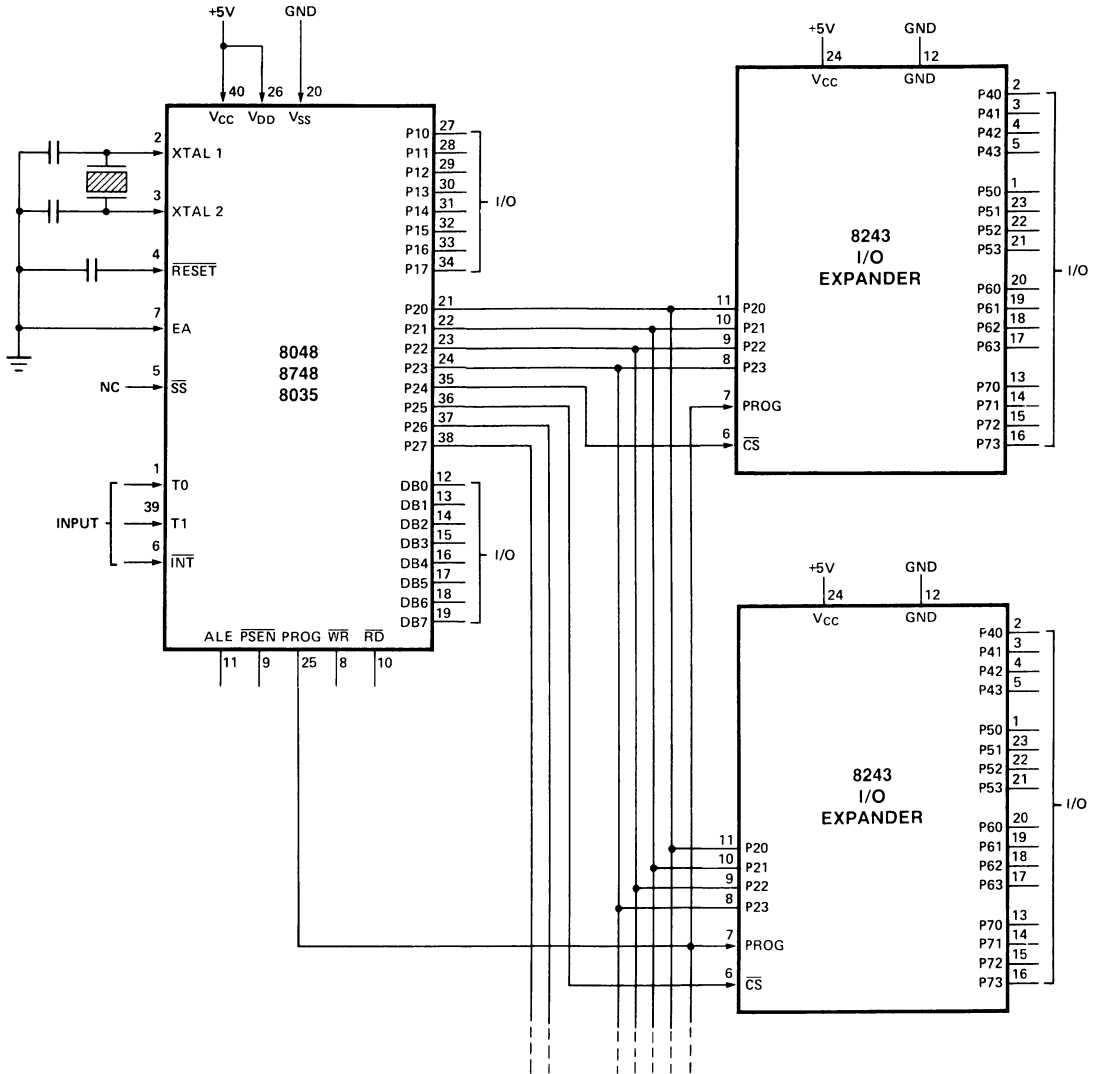
## APPLICATION EXAMPLES



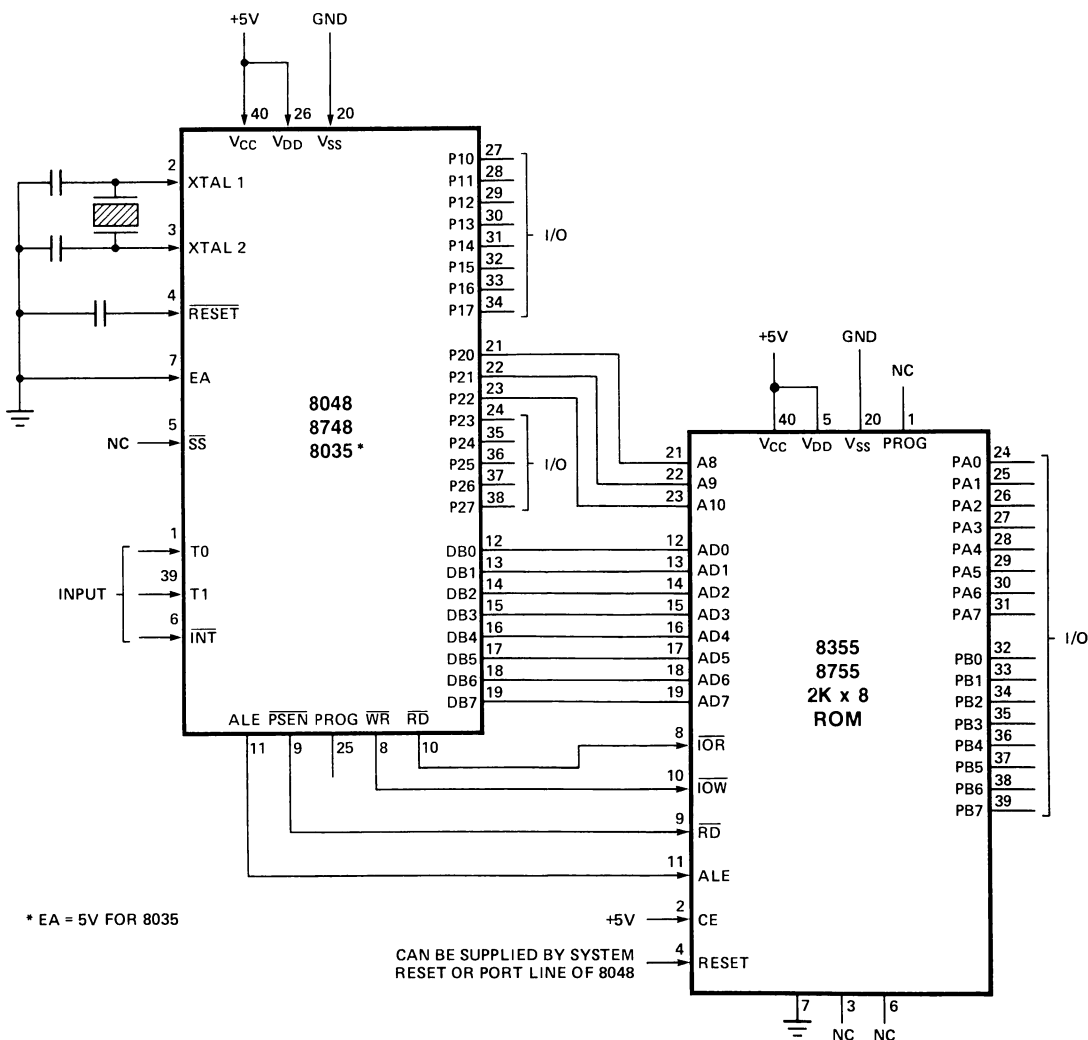
### ADDING AN I/O EXPANDER



## APPLICATION EXAMPLES



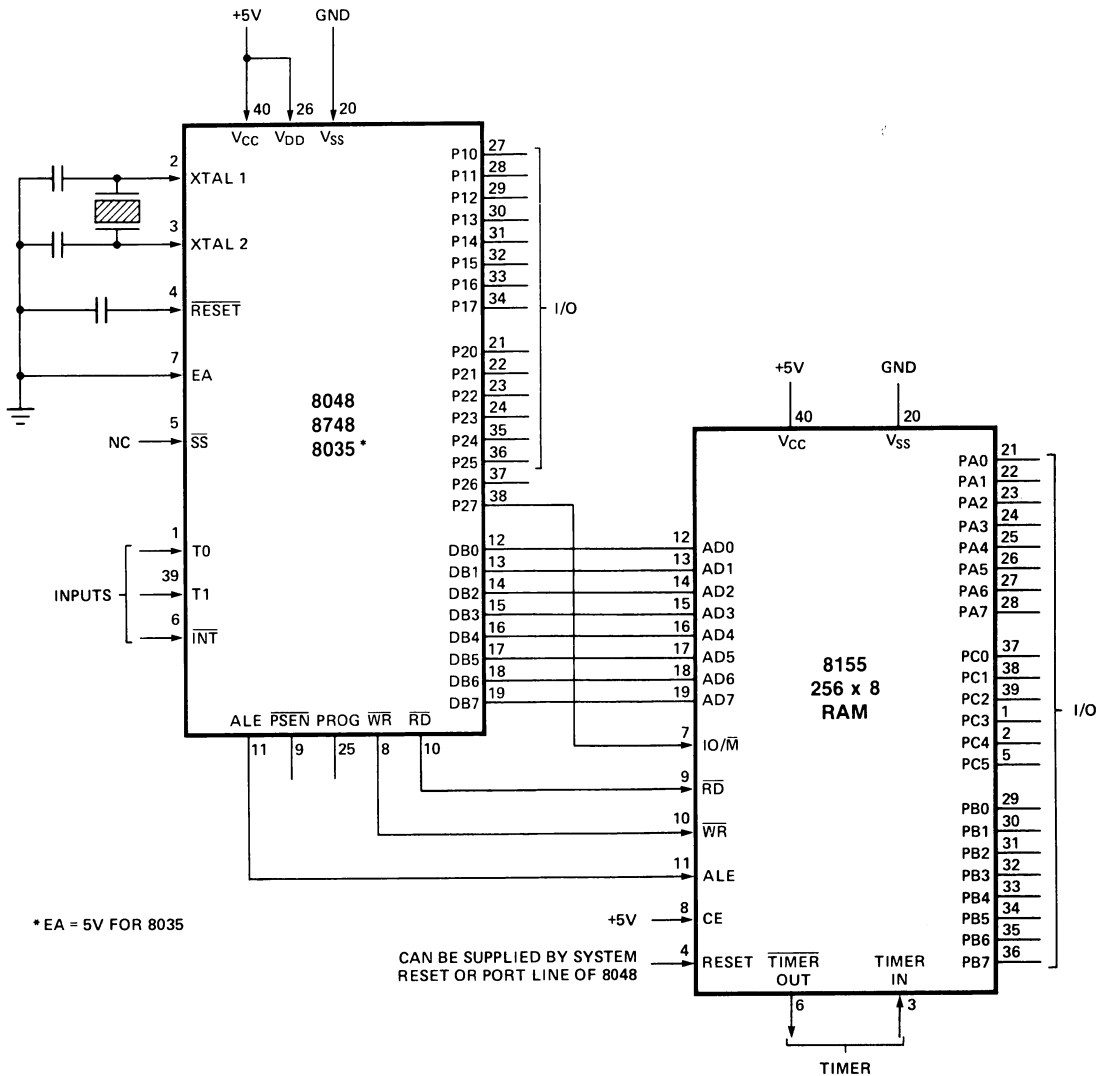
## ADDING MULTIPLE I/O EXPANDERS



- External I/O parts are addressed as data memory PA=00 PB=01
- If the 8048's internal Program Memory is used this configuration will result in the upper 1K of external memory being addressed before the lower 1K. Inverting A10 will correct this if necessary.

## ADDING A PROGRAM MEMORY AND I/O EXPANDER

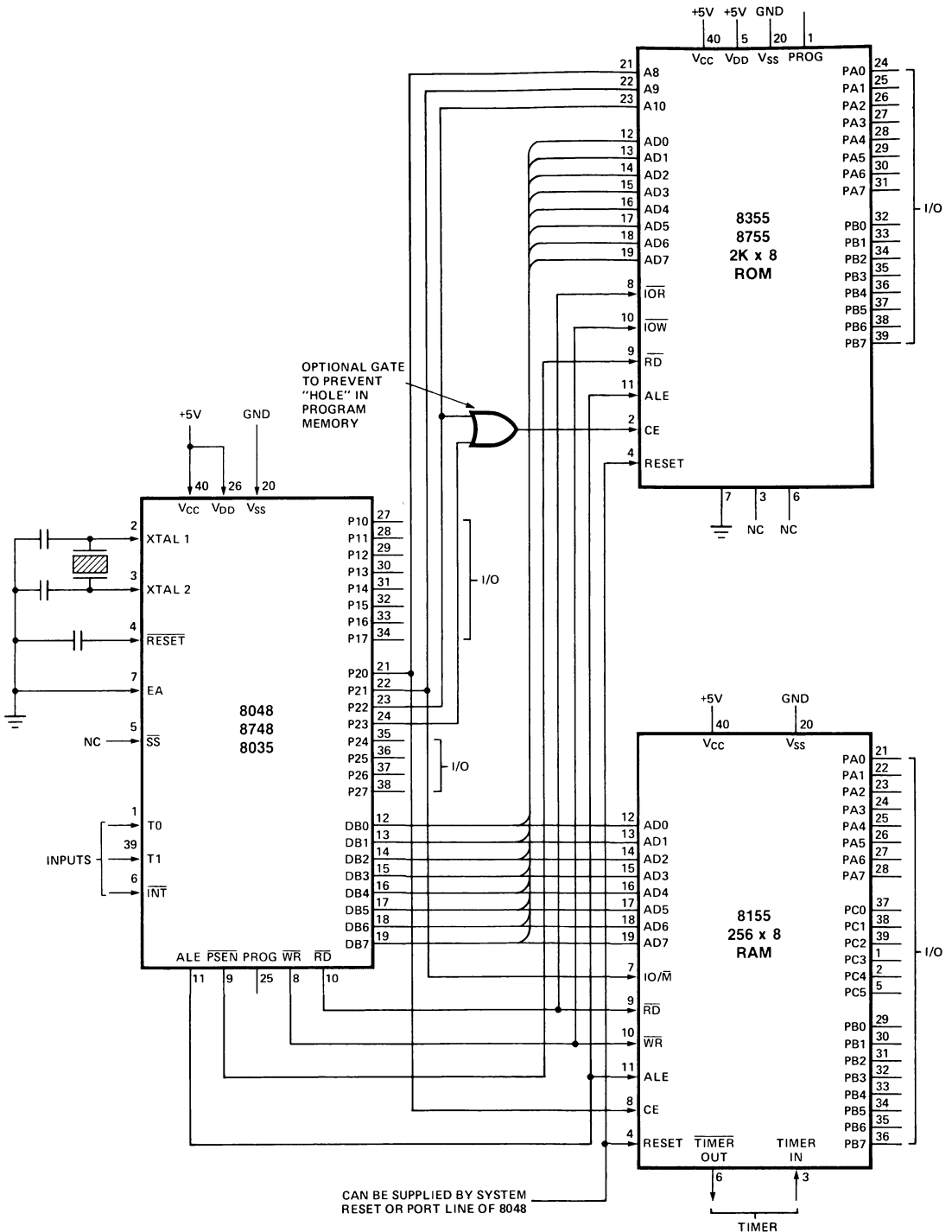
## APPLICATION EXAMPLES



- Both I/O and RAM are addressed as data memory
- Writing a bit to P27 determines whether RAM or I/O is to be accessed

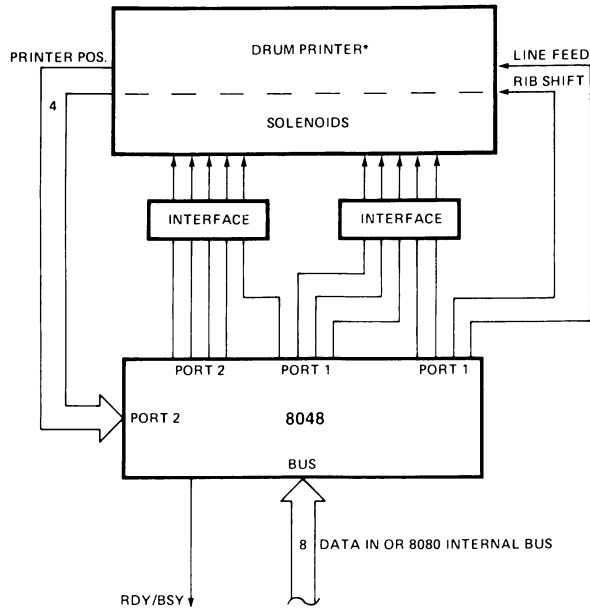
## ADDING A DATA MEMORY AND I/O EXPANDER

## APPLICATION EXAMPLES



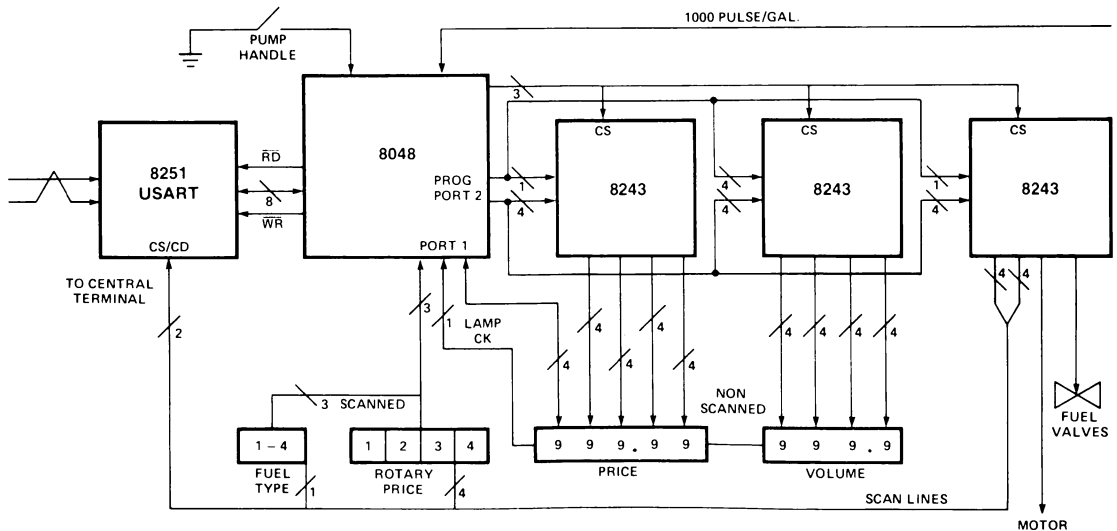
- This configuration is explained in section 3.4

## APPLICATION EXAMPLES



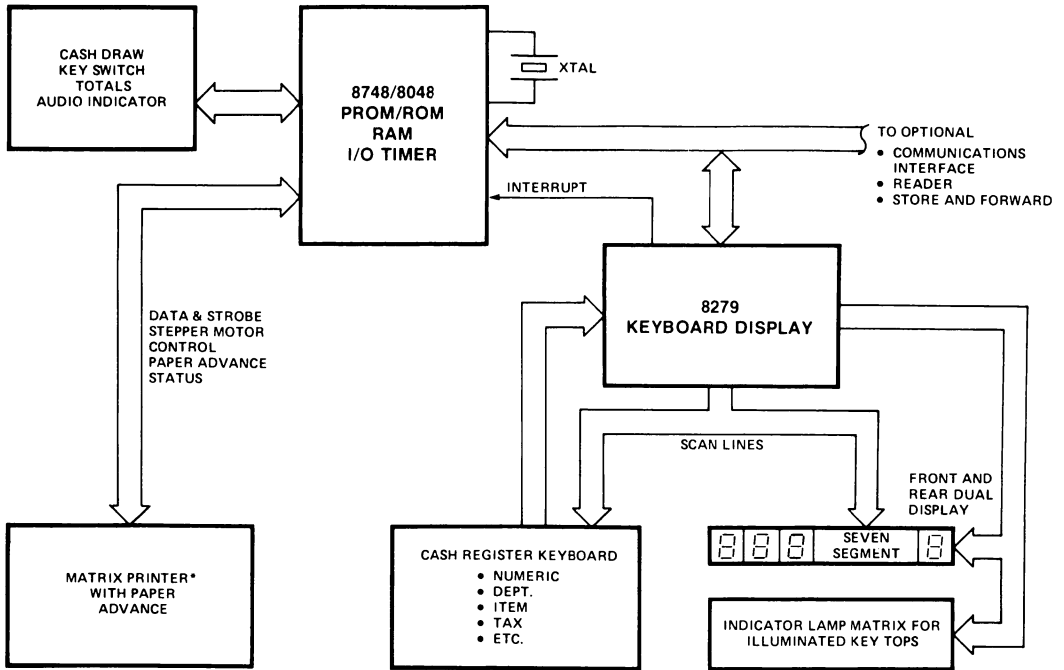
\*SEIKO  $\mu$ 101

### 8048 INTERFACE TO DRUM PRINTER



### MCS-48™ GAS PUMP

## APPLICATION EXAMPLES



\*DRUM PRINTER MAY BE USED.  
DRUM PRINTER REQUIRES  
MORE OUTPUTS WHICH CAN BE  
OBTAINED FROM AN EXPANDER  
DEVICE.

## LOW COST POINT OF SALE TERMINAL

## 5.2 Software Examples

The following routines are written as subroutines. R0 and R1 are used as data pointers, R2 is used as an extension of the accumulator and R3 is used as a loop counter.

RX0 = R0  
AEX = R2

### DOUBLE ADD

```
DADD:  DEC    RX0      ;GET LOW BYTE AND ADD TO A
        ADD    A,@RX0
        INC    RX0      ;GET HI BYTE AND ADD TO AEX
        XCH    A,AEX
        ADDC   A,@RX0
        XCH    A,AEX
        RET                      ;RETURN
```

### DOUBLE SUBTRACT

```
DMIN:  DEC    RX0      ;GET LOW BYTE AND SUB FROM A
        CPL    A
        ADD    A,@RX0
        CPL    A
        INC    RX0      ;GET HI BYTE AND SUB FROM AEX
        XCH    A,AEX
        CPL    A
        ADDC   A,@RX0
        CPL    A
        XCH    A,AEX
        RET                      ;RETURN
```

### DOUBLE LOAD

```
DLD:   DEC    RX0      ;GET LOW BYTE AND PLACE IN A
        MOV    A,@RX0
        INC    RX0      ;GET HI BYTE AND PLACE IN AEX
        XCH    A,AEX
        MOV    A,@RX0
        XCH    A,AEX
        RET                      ;RETURN
```

### DOUBLE STORE

```
DST:   DEC    RX0      ;MOVE A INTO LOW BYTE
        MOV    @RX0,A
        INC    RX0      ;MOVE AEX INTO HIGH BYTE
        XCH    A,AEX
        MOV    @RX0,A
        XCH    A,AEX
        RET                      ;RETURN
```

**DOUBLE EXCHANGE**

```
DEX:  DEC    RX0      ;EXCHANGE A AND LOW BYTE
      XCH    A,@RX0
      INC    RX0      ;EXCHANGE AEX AND HIGH BYTE
      XCH    A,AEX
      XCH    A,@RX0
      XCH    A,AEX
      RET                      ;RETURN
```

**DOUBLE LEFT LOGICAL SHIFT**

```
LLSH:  RLC    A        ;SHIFT A
      XCH    A,AEX     ;SHIFT AEX
      RLC    A
      XCH    A,AEX
      RET                      ;RETURN
```

**DOUBLE RIGHT LOGICAL SHIFT**

```
RLSH:  XCH    A,AEX     ;SHIFT AEX
      RRC    A
      XCH    A,AEX
      RRC    A        ;SHIFT A
      RET                      ;RETURN
```

**DOUBLE RIGHT ARITHMETIC SHIFT**

```
RASH:  CLR    C        ;SET CARRY
      CPL    C

      XCH    A,AEX     ;IF AEX[7]<>1 THEN
      JB7    $+3
      CLR    C        ;CLEAR CARRY
      RRC    A        ;SHIFT C INTO AEX
      XCH    A,AEX
      RRC    A        ;SHIFT A
      RET                      ;RETURN
```

**SINGLE PRECISION BINARY MULTIPLY**

This routine assumes a one-byte multiplier and a one-byte multiplicand. The product, therefore, is two-bytes long.

The algorithm follows these steps:

1. The registers are arranged as follows:

ACC — 0  
R1 — Multiplier  
R2 — Multiplicand  
R3 — Loop Counter (=8)

The Accumulator and register R1 are treated as a register pair when they are shifted right (see Step 2)

2. The Accumulator and R1 are shifted right one place, thus the LSB of the multiplier goes into the carry.
3. The multiplicand is added to the accumulator if the carry bit is a 'one'. No action if the carry is a 'zero'.
4. Decrement the loop counter and loop (return to Step 2) until it reaches zero.
5. Shift the result right one last time just before exiting the routine

\*The result will be found in the Accumulator (MS Byte) and R1 (LS Byte).



## BINARY MULTIPLY

```

BMPY:    MOV    R3,#08H    ;SET COUNTER TO 8
          CLR    A          ;CLEAR A
          CLR    C          ;CLEAR CARRY BIT

BMPI:    RRC     A          ;DOUBLE SHIFT RIGHT ACC & R1
          XCH    A,R1       ;INTO CARRY
          RRC     A
          XCH    A,R1
          JNC     BMP3      ;IF CARRY=1 ADD, OTHERWISE DON'T
          ADD     A,R2      ;ADD MULTIPLICAND TO ACCUMULATOR

BMP3:    DJNZ    R3,BMPI    ;DECREMENT COUNTER AND LOOP IF 0
          RRC     A          ;DO A FINAL RIGHT SHIFT AT THE
          XCH    A,R1       ;END OF THE ROUTINE
          RRC     A
          XCH    A,R1
  
```

## INTERRUPT HANDLING

This interrupt routine assumes single level interrupt. The purpose is to store the status of the machine at the time the interrupt occurs by storing contents of all registers, accumulator, and the status word. At the end of the interrupt the state of the machine is restored and interrupts are enabled again.

```

INTRPT:  SEL     RB1        ;SAVE WORKING REGISTERS
          MOV     @R0,A      ;R0 IN ALTERNATE REGISTER
                               ;BANK CONTAINS SACC
                               ;POINTER FOR SAVING
                               ;ACCUMULATOR

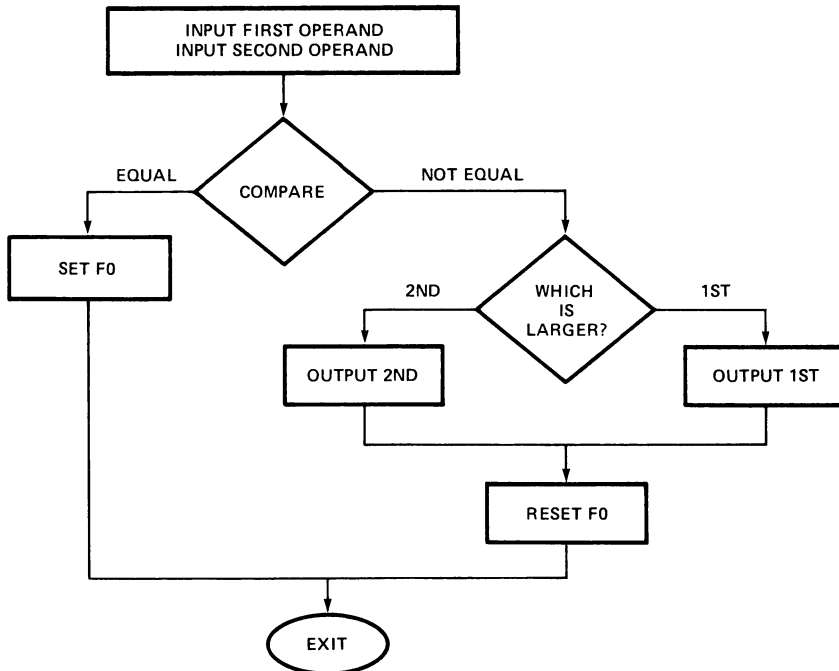
          |         |      { INTERRUPT SERVICE
          |         |      { ROUTINE

          MOV     R0,SACC    ;RESTORE SACC
          MOV     A,@R0      ;RESTORE ACCUMULATOR
          RETR      ;RESTORE WORKING REGISTERS
                               ;RESTORE PSW AND
                               ;RE-ENABLE INTERRUPTS
  
```

## 2 BYTE PROCESSING SYSTEM

A suggested model of a processing routine takes two single byte inputs from different ports, compares them, and performs the following, depending on the result of the comparison:

(If Equal) Sets Flag and Exits  
(If Not Equal) Resets Flag and Outputs the Larger to a Third Port



```

PROCESS: CLR    F0      ;CLEAR F0 BIT (INITIALIZE)
          IN      A,P1   ;READ FIRST INPUT, STORE IN R0
          MOV     R0,A
          IN      A,P2   ;READ SECOND INPUT, STORE IN R1
          MOV     R1,A
          CPL     A      ;SUBTRACT SECOND FROM FIRST
          INC     A      ;(2's COMPLEMENT AND ADD)
          ADD     A,R0
          JZ      EQU    ;BRANCH IF THEY ARE EQUAL
          JN      SECOND ;IF NEGATIVE, SECOND WAS LARGER
          MOV     A,R0    ;ELSE, OUTPUT FIRST
          OUTL    P3,A
          JMP     DONE   ;EXIT

SECOND: MOV     A,R1     ;OUTPUT SECOND
          OUTL    P3,A
          JMP     DONE   ;EXIT

EQU:    CPL     F0      ;SET F0
          JMP     DONE   ;EXIT
  
```

**A/D CONVERTER**

An A/D converter can be constructed from a D/A converter, a comparator op-amp and a short software routine that performs successive approximation.

The processor sends 8-bits of data out to the DAC via an output port. The output of the DAC is compared to the analog input being converted. The result of the comparison (0 if

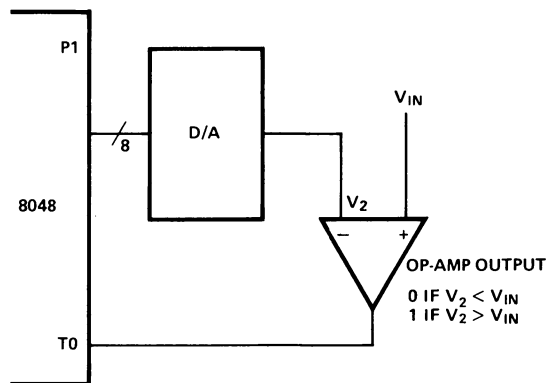
lower, 1 if higher) then goes back into the processor for handling either via an input port or an input line that sets a flag. This all allows the processor to estimate the proper digital representation of the analog input by first typing the MSB — and keeping it if the input says 'too low still' or dropping it if the input says 'too high now'. From there each bit in order of significance is tried and either kept or discarded.

```

MOV     R7,#08H   ;COUNTER R7=8
CLR     A         ;CLEAR A, R5, R6
MOV     R5,A
MOV     R6,A
CLR     C         ;SET CARRY
CPL     C

LOOP:   MOV     A,R5   ;MOVE TEST BIT RIGHT
        RRC     A     ;FROM MSB TO LSB
        MOV     R5,A
        ORL     A,R6   ;ADD IT TO PRESENT VALUE IN R6
        OUTL    P1,A
        JNT0    NOPE   ;TEST THAT NEW VALUE
                        ;IF FLAG IS HIGH NEW VALUE TOO LARGE
        MOV     R6,A   ;IF FLAG LOW, NEW VALUE RETAINED
NOPE:   DJNZ    R7,LOOP ;GO ON TO NEXT BIT

```



AMERICAN MEDICAL ASSOCIATION  
PUBLISHED WEEKLY  
CHICAGO, ILL., U.S.A.  
Subscription price, Five Dollars Per Annum in Advance  
Single Copies, Fifteen Cents  
Entered as Second-Class Matter, May 2, 1902  
Postpaid  
Acceptance for mailing at special rate of postage provided for in Act of October 3, 1917  
Authorized by Act of October 3, 1917  
Copyright, 1918, by American Medical Association  
Printed at the American Medical Association, 535 North Dearborn Street, Chicago, Ill.

CONTENTS

Original Articles	1
Editorial	1
Book Reviews	1
Correspondence	1
Obituary	1
Announcements	1
Index	1

ADVERTISING

Advertisements are accepted for insertion on the following basis:  
One insertion, 10 cents per line  
One month, \$3.00 per line  
Three months, \$8.00 per line  
Six months, \$15.00 per line  
One year, \$28.00 per line  
Special rates for long term contracts  
Illustrations, 50 cents per illustration  
Color illustrations, \$1.00 per illustration  
Readers' Service, 50 cents per year  
Subscription Service, 50 cents per year  
Advertising Manager, American Medical Association, 535 North Dearborn Street, Chicago, Ill.

## Chapter 6

# MCS-48™ COMPONENT SPECIFICATIONS



## MCS-48™ COMPONENT SPECIFICATIONS

8048	ROM Microcomputer .....	6-1
8748	EPROM Microcomputers .....	6-1
8035	Microcomputers .....	6-1
8355	ROM and I/O Expander .....	6-7
8755	EPROM and I/O Expander .....	6-13
8155	RAM and I/O Expander .....	6-19
8243	MCS-48™ I/O Expander .....	6-29



## 8048/8748/8035

# SINGLE COMPONENT 8-BIT MICROCOMPUTER

**\*8048 Mask Programmable ROM**

**\*8748 User Programmable/Erasable EPROM**

**\*8035 External ROM or EPROM**

- 8-Bit CPU, ROM, RAM, I/O in Single Package
- Interchangeable ROM and EPROM Versions
- Single 5V Supply
- 2.5  $\mu$ sec and 5.0  $\mu$ sec Cycle Versions  
All Instructions 1 or 2 Cycles.
- Over 90 Instructions: 70% Single Byte
- 1K x 8 ROM/EPROM
- 64 x 8 RAM
- 27 I/O Lines
- Interval Timer/Event Counter
- Easily Expandable Memory and I/O
- Compatible with MCS-80™ Peripherals
- Single Level Interrupt

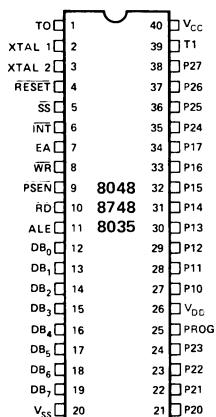
The Intel® 8048/8748/8035 is a totally self-sufficient 8-bit parallel computer fabricated on a single silicon chip using Intel's N-channel silicon gate MOS process.

The 8048 contains a 1K x 8 program memory, a 64 x 8 RAM data memory, 27 I/O lines, and an 8-bit timer/counter in addition to on board oscillator and clock circuits. For systems that require extra capability, the 8048 can be expanded using standard memories and MCS-80™ (8080A) peripherals. The 8035 is the equivalent of an 8048 without program memory.

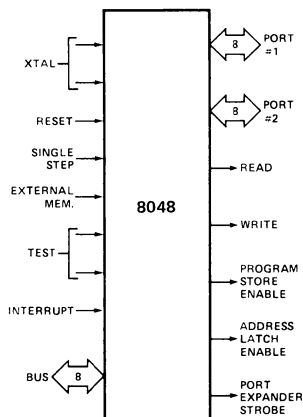
To reduce development problems to a minimum and provide maximum flexibility, three interchangeable pin-compatible versions of this single component microcomputer exist: the 8748 with user-programmable and erasable EPROM program memory for prototype and preproduction systems, the 8048 with factory-programmed mask ROM program memory for low-cost high volume production, and the 8035 without program memory for use with external program memories.

This microprocessor is designed to be an efficient controller as well as an arithmetic processor. The 8048 has extensive bit handling capability as well as facilities for both binary and BCD arithmetic. Efficient use of program memory results from an instruction set consisting mostly of single byte instructions and no instructions over two bytes in length.

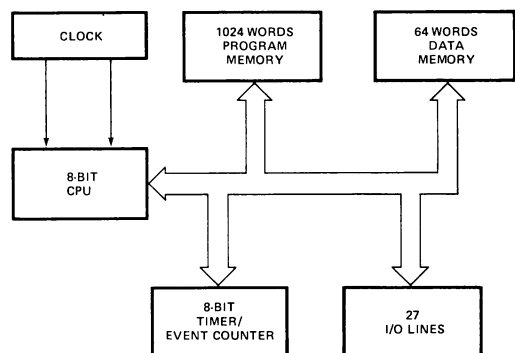
### PIN CONFIGURATION



### LOGIC SYMBOL



### BLOCK DIAGRAM



## ABSOLUTE MAXIMUM RATINGS\*

Ambient Temperature Under Bias ..... 0°C to 70°C  
 Storage Temperature ..... -65°C to +150°C  
 Voltage On Any Pin With Respect  
     to Ground ..... -0.5V to +7V  
 Power Dissipation ..... 1.5 Watt

### \*COMMENT:

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

## D.C. AND OPERATING CHARACTERISTICS $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ , $V_{CC} = V_{DD} = +5V \pm 5\%$ , $V_{SS} = 0V$

Symbol	Parameter	Limits			Unit	Test Conditions
		Min.	Typ.	Max.		
$V_{IL}$	Input Low Voltage (All Except XTAL1, XTAL2)	-.5		.8	V	
$V_{IH}$	Input High Voltage (All Except XTAL1, XTAL2, RESET)	2.0		$V_{CC}$	V	
$V_{IH1}$	Input High Voltage (RESET, XTAL1)	3.0		$V_{CC}$	V	
$V_{OL}$	Output Low Voltage (BUS, RD, WR, PSEN, ALE)			.45	V	$I_{OL} = 2.0\text{mA}$
$V_{OL1}$	Output Low Voltage (All Other Outputs Except PROG)			.45	V	$I_{OL} = 1.6\text{mA}$
$V_{OH}$	Output High Voltage (BUS, RD, WR, PSEN, ALE)	2.4			V	$I_{OH} = 100\mu\text{A}$
$V_{OH1}$	Output High Voltage (All Other Outputs)	2.4			V	$I_{OH} = 50\mu\text{A}$
$I_{IL}$	Input Leakage Current (T1, EA, INT)			$\pm 10$	$\mu\text{A}$	$V_{SS} \leq V_{IN} \leq V_{CC}$
$I_{OL}$	Output Leakage Current (Bus, T0) (High Impedance State)			-10	$\mu\text{A}$	$V_{CC} \geq V_{IN} \geq V_{SS} + .45$
$I_{DD}$	$V_{DD}$ Supply Current			30	mA	
$I_{CC}$	$V_{CC}$ Supply Current			180	mA	

## A.C. CHARACTERISTICS $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ , $V_{CC} = V_{DD} = +5V \pm 5\%$ , $V_{SS} = 0V$

Symbol	Parameter	8048/8748/8035		8048-8 8748-8 8035-8		Unit	Conditions
		Min.	Max.	Min.	Max.		
$t_{LL}$	ALE Pulse Width	400		800		ns	
$t_{AL}$	Address Setup to ALE	150		150		ns	
$t_{LA}$	Address Hold from ALE	80		80		ns	
$t_{CC}$	Control Pulse Width (PSEN, RD, WR)	900		1800		ns	
$t_{DW}$	Data Set-Up Before WR	500		1000		ns	
$t_{WD}$	Data Hold After WR	80		80		ns	$C_L = 20\text{pF}$
$t_{CY}$	Cycle Time	2.5		5.0		$\mu\text{s}$	6 MHz XTAL (3 MHz XTAL for -8)
$t_{DR}$	Data Hold	0	130	0	130	ns	
$t_{RD}$	PSEN, RD to Data In		500		1000	ns	
$t_{AW}$	Address Setup to WR	230		260		ns	
$t_{AD}$	Address Setup to Data In		950		1900	ns	
$t_{AFC}$	Address Float to RD, PSEN	0		0		ns	

## A.C. TEST CONDITIONS

Control Outputs:  
 BUS Outputs:

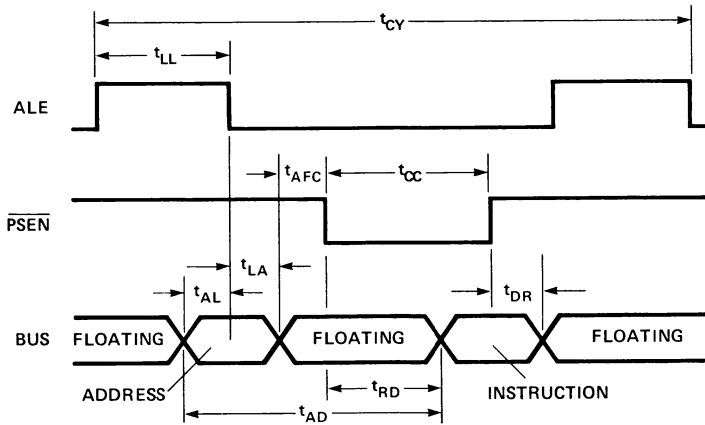
$C_L = 80\text{ pF}$ , 2.2K to  $V_{SS}$ , 4.3K to  $V_{CC}$   
 $C_L = 150\text{ pF}$ , 2.2K to  $V_{SS}$ , 4.3K to  $V_{CC}$

$t_{CY} = 2.5\mu\text{s}$

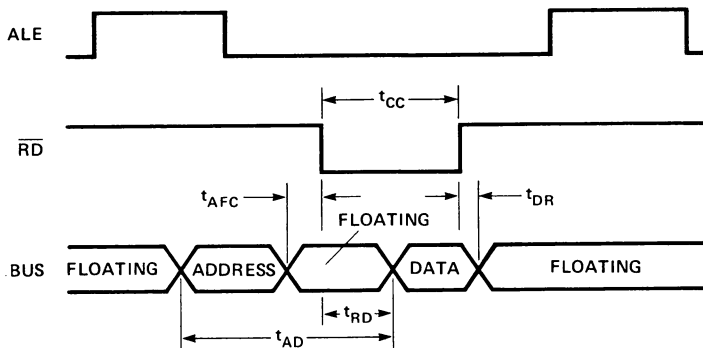


## WAVEFORMS

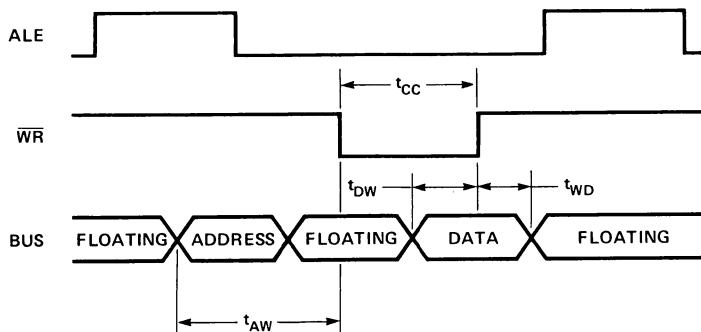
## INSTRUCTION FETCH FROM EXTERNAL PROGRAM MEMORY



## READ FROM EXTERNAL DATA MEMORY



## WRITE TO EXTERNAL DATA MEMORY



## PIN DESCRIPTION

Designation	Pin #	Function	Designation	Pin #	Function
V <sub>SS</sub>	20	Circuit GND potential	$\overline{RD}$	10	Output strobe activated during a BUS read. Can be used to enable data onto the BUS from an external device.  Used as a Read Strobe to External Data Memory. (Active low)
V <sub>DD</sub>	26	Programming power supply; +25V during program, +5V during operation for both ROM and PROM. Low power standby pin in 8048 ROM version.	$\overline{RESET}$	4	Input which is used to initialize the processor. Also used during PROM programming verification, and power down. (Active low)
V <sub>CC</sub>	40	Main power supply; +5V during operation and programming.	$\overline{WR}$	8	Output strobe during a BUS write. (Active low) (Non TTL V <sub>IH</sub> )  Used as write strobe to External Data Memory.
PROG	25	Program pulse (+25V) input pin during 8748 programming.  Output strobe for 8243 I/O expander.	ALE	11	Address Latch Enable. This signal occurs once during each cycle and is useful as a clock output.  The negative edge of ALE strobes address into external data and program memory.
P10-P17 Port 1	27-34	8-bit quasi-bidirectional port.	$\overline{PSEN}$	9	Program Store Enable. This output occurs only during a fetch to external program memory. (Active low)
P20-P27 Port 2	21-24 35-38	8-bit quasi-bidirectional port.  P20-P23 contain the four high order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243	$\overline{SS}$	5	Single step input can be used in conjunction with ALE to "single step" the processor through each instruction. (Active low)
D0-D7 BUS	12-19	True bidirectional port which can be written or read synchronously using the $\overline{RD}$ , $\overline{WR}$ strobes. The port can also be statically latched.  Contains the 8 low order program counter bits during an external program memory fetch, and receives the addressed instruction under the control of PSEN. Also contains the address and data during an external RAM data store instruction, under control of ALE, $\overline{RD}$ , and $\overline{WR}$ .	EA	7	External Access input which forces all program memory fetches to reference external memory. Useful for emulation and debug, and essential for testing and program verification. (Active high)
T0	1	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 can be designated as a clock output using ENT0 CLK instruction. T0 is also used during programming.	XTAL1	2	One side of crystal input for internal oscillator. Also input for external source. (Not TTL Compatible)
T1	39	Input pin testable using the JT1, and JNT1 instructions. Can be designated the timer/counter input using the STRT CNT instruction.	XTAL2	2	Other side of crystal input.
$\overline{INT}$	6	Interrupt input. Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. Also testable with conditional jump instruction. (Active low)			

## INSTRUCTION SET

	Mnemonic	Description	Bytes	Cycle		Mnemonic	Description	Bytes	Cycles
Accumulator	ADD A, R	Add register to A	1	1	Subroutine	CALL	Jump to subroutine	2	2
	ADD A, @R	Add data memory to A	1	1		RET	Return	1	2
	ADD A, #data	Add immediate to A	2	2		RETR	Return and restore status	1	2
	ADDC A, R	Add register with carry	1	1					
	ADDC A, @R	Add data memory with carry	1	1					
	ADDC A, #data	Add immediate with carry	2	2	Flags	CLR C	Clear Carry	1	1
	ANL A, R	And register to A	1	1		CPL C	Complement Carry	1	1
	ANL A, @R	And data memory to A	1	1		CLR F0	Clear Flag 0	1	1
	ANL A, #data	And immediate to A	2	2		CPL F0	Complement Flag 0	1	1
	ORL A, R	Or register to A	1	1		CLR F1	Clear Flag 1	1	1
	ORL A, @R	Or data memory to A	1	1		CPL F1	Complement Flag 1	1	1
	ORL A, #data	Or immediate to A	2	2	Data Moves	MOV A, R	Move register to A	1	1
	XRL A, R	Exclusive Or register to A	1	1		MOV A, @R	Move data memory to A	1	1
	XRL A, @R	Exclusive or data memory to A	1	1		MOV A, #data	Move immediate to A	2	2
	XRL A, #data	Exclusive or immediate to A	2	2		MOV R, A	Move A to register	1	1
	INC A	Increment A	1	1		MOV @R, A	Move A to data memory	1	1
	DEC A	Decrement A	1	1		MOV R, #data	Move immediate to register	2	2
	CLR A	Clear A	1	1		MOV @R, #data	Move immediate to data memory	2	2
	CPL A	Complement A	1	1		MOV A, PSW	Move PSW to A	1	1
	DA A	Decimal Adjust A	1	1		MOV PSW, A	Move A to PSW	1	1
	SWAP A	Swap nibbles of A	1	1		XCH A, R	Exchange A and register	1	1
	RL A	Rotate A left	1	1		XCH A, @R	Exchange A and data memory	1	1
	RLC A	Rotate A left through carry	1	1		XCHD A, @R	Exchange nibble of A and register	1	1
	RR A	Rotate A right	1	1		MOVX A, @R	Move external data memory to A	1	2
	RRC A	Rotate A right through carry	1	1		MOVX @R, A	Move A to external data memory	1	2
						MOV P A, @A	Move to A from current page	1	2
						MOV P3 A, @A	Move to A from Page 3	1	2
Input/Output	IN A, P	Input port to A	1	2	Timer/Counter	MOV A, T	Read Timer/Counter	1	1
	OUTL P, A	Output A to port	1	2		MOV T, A	Load Timer/Counter	1	1
	ANL P, #data	And immediate to port	2	2		STRT T	Start Timer	1	1
	ORL P, #data	Or immediate to port	2	2		STRT CNT	Start Counter	1	1
	INS A, BUS	Input BUS to A	1	2		STOP TCNT	Stop Timer/Counter	1	1
	OUTL BUS, A	Output A to BUS	1	2		EN TCNTI	Enable Timer/Counter Interrupt	1	1
	ANL BUS, #data	And immediate to BUS	2	2		DIS TCNTI	Disable Timer/Counter Interrupt	1	1
	ORL BUS, #data	Or immediate to BUS	2	2	Control	EN I	Enable external interrupt	1	1
	MOVD A, P	Input Expander port to A	1	2		DIS I	Disable external interrupt	1	1
	MOVD P, A	Output A to Expander port	1	2		SEL RB0	Select register bank 0	1	1
Registers	ANLD P, A	And A to Expander port	1	2		SEL RB1	Select register bank 1	1	1
	ORLD P, A	Or A to Expander port	1	2		SEL MB0	Select memory bank 0	1	1
Branch	INC R	Increment register	1	1		SEL MB1	Select memory bank 1	1	1
	INC @R	Increment data memory	1	1		ENT0 CLK	Enable Clock output on T0	1	1
	DEC R	Decrement register	1	1	NOP	NOP	No Operation	1	1
	JMP addr	Jump unconditional	2	2					
	JMPP @A	Jump indirect	1	2					
	DJNZ R, addr	Decrement register and skip	2	2					
	JC addr	Jump on Carry = 1	2	2					
	JNC addr	Jump on Carry = 0	2	2					
	JZ addr	Jump on A Zero	2	2					
	JNZ addr	Jump on A not Zero	2	2					
	JT0 addr	Jump on T0 = 1	2	2					
	JNT0 addr	Jump on T0 = 0	2	2					
	JT1 addr	Jump on T1 = 1	2	2					
	JNT1 addr	Jump on T1 = 0	2	2					
	JF0 addr	Jump on F0 = 1	2	2					
	JF1 addr	Jump on F1 = 1	2	2					
	JTF addr	Jump on timer flag	2	2					
	JNI addr	Jump on $\overline{INT}$ = 0	2	2					
	JBb addr	Jump on Accumulator Bit	2	2					

Mnemonics copyright Intel Corporation 1976



# 8355

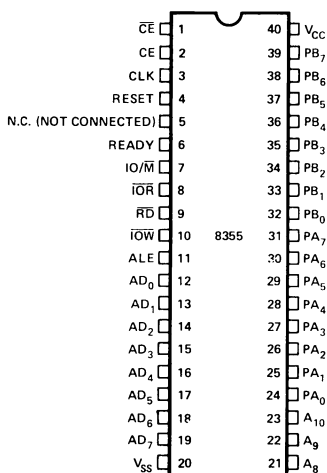
## ROM AND I/O EXPANDER

- 2K x 8 ROM
- 2 Eight Bit I/O Ports
- Internal Address Latch
- I/O Lines Individually Assignable as Input or Output
- Single 5V Supply
- 40 Pin DIP
- Completely Interchangeable With 8755 EPROM

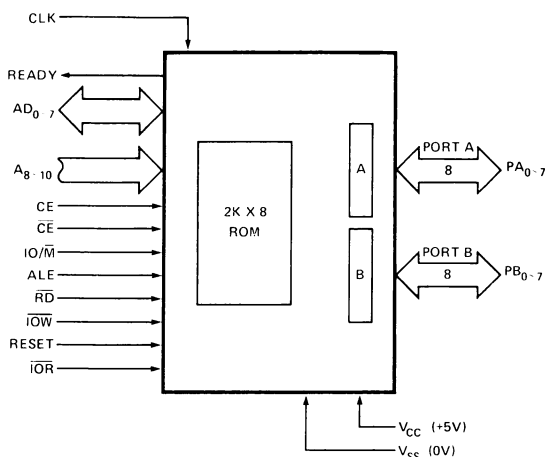
The 8355 is designed to expand both the program memory and I/O capability of the MCS-48™ single component microcomputers (the 8748, 8048 and 8035). This expander increases program memory by 2K words and adds 16 I/O lines to the basic microcomputer without the necessity of any additional components. The completely interchangeable 8755 light erasable EPROM and 8355 mask programmed ROM provide a simple transition from prototype to production. Both versions operate from a single 5V supply and are totally speed compatible with the MCS-48 microcomputers.

The 16 I/O lines are addressed as 2 eight bit I/O ports, yet single lines can be individually designated as input or as output under software control. Outputs are double buffered to prevent any output glitches.

### PIN CONFIGURATION



### BLOCK DIAGRAM



## 8355 FUNCTIONAL PIN DEFINITION

Symbol	Function	Symbol	Function
ALE	When ALE (Address Latch Enable) is high, AD <sub>0-7</sub> , IO/ $\overline{M}$ , A <sub>8-10</sub> , and $\overline{CE}$ enter address latches. The signals (AD, IO/ $\overline{M}$ , A <sub>8-10</sub> , $\overline{CE}$ ) are latched in at the trailing edge of ALE.	CLK	The CLK is used to force the READY into its high impedance state after it has been forced low by $\overline{CE}$ low and ALE high.
AD <sub>0-7</sub>	Bi-directional Address/Data bus. The lower 8-bits of the ROM or I/O address are applied to the bus lines when ALE is high.  During an I/O cycle, Port A or B are selected based on the latched value of AD <sub>0</sub> . If $\overline{RD}$ or $\overline{IOR}$ is low when latched $\overline{CE}$ is low, the output buffers present data on the bus.	READY	Ready is an tri-state output controlled by $\overline{CE}$ , ALE and CLK. READY is forced low by $\overline{CE}$ during the time ALE is high, and remains low until the rising edge of the next CLK (see Figure 4).
A <sub>8-10</sub>	These are the high order bits of the ROM address. They do not affect I/O operations.	PA <sub>0-7</sub>	These are general purpose I/O pins. Their input/output direction is determined by the contents of Data Direction Register (DDR). Port A is selected for write operations by $\overline{CE}$ and $\overline{IOW}$ low and a 0 previously latched from AD <sub>0</sub> .
$\overline{CE}$ CE	When the latched $\overline{CE}$ is high or latched CE is low, no read or write operation will occur. The AD <sub>0-7</sub> and READY outputs will go into their high impedance state.	PB <sub>0-7</sub>	Read operation is selected by either $\overline{IOR}$ low or IO/ $\overline{M}$ high and $\overline{RD}$ low, and the latched $\overline{CE}$ low and AD <sub>0</sub> low.
IO/ $\overline{M}$	If the latched IO/ $\overline{M}$ is high when $\overline{RD}$ is low, the output data comes from an I/O port. If it is low the output data comes from the ROM.	RESET	This general purpose I/O port is identical to Port A except that it is selected by a 1 latched from AD <sub>0</sub> .  An input high on RESET causes all pins in Ports A and B to assume input mode.
$\overline{RD}$	If the latched $\overline{CE}$ is low when $\overline{RD}$ goes low, the AD <sub>0-7</sub> output buffers are enabled and output either the selected ROM location or I/O port. When both $\overline{RD}$ and $\overline{IOR}$ are high, the AD <sub>0-7</sub> output buffers are tri-stated.	$\overline{IOR}$	When $\overline{CE}$ is low, a low on $\overline{IOR}$ will output the selected I/O port onto the AD bus. $\overline{IOR}$ low performs the same function as the combination IO/ $\overline{M}$ high and $\overline{RD}$ low.
$\overline{IOW}$	If the latched $\overline{CE}$ is low, a low on $\overline{IOW}$ causes the output port pointed to by the latched value of AD <sub>0</sub> to be written with the data on AD <sub>0-7</sub> . The state of IO/ $\overline{M}$ is ignored.	VCC	+5 volt supply.
		VSS	0 volt supply.

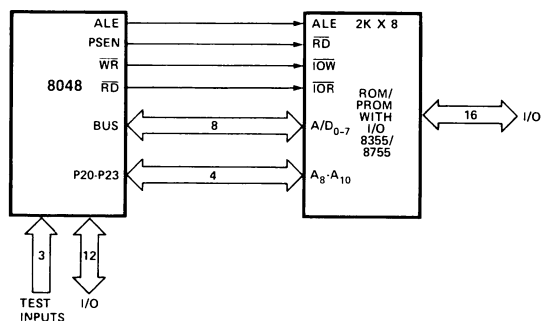
## FUNCTIONAL DESCRIPTION

**Program Memory** — The 8355 contains an 8-bit address latch which allows it to interface directly to MCS-48 Microcomputers without additional hardware. Program memory is accessed by applying 11 bits of address to the  $A_0 - A_{10}$  inputs and a low level on the  $IO/\overline{M}$  and  $\overline{CE}$  inputs then latching these inputs with ALE. The  $\overline{CE}$  input serves to select one of several possible 8355s in a system and the  $IO/\overline{M}$  signal indicates that a subsequent read operation will be from program memory. While ALE is high the  $A_0 - A_{10}$ ,  $IO/\overline{M}$ , and  $\overline{CE}$  inputs are allowed into the 8355 and when ALE is brought low, these inputs are latched. If the latched conditions indicate that a program memory fetch is to occur, a low level on  $\overline{RD}$  will cause the data to be outputted on the data bus.

**I/O Ports** — The I/O lines are organized as two 8-bit static ports which can be read or written using the **IOR** and **IOW** control lines. Associated with each port is an 8-bit Data Direction Register (DDR) which serves to define each of the 8 lines of the port as either an input or an output. A “1” bit in the DDR sets the corresponding port bit to the output mode while a “0” designates the input mode. The two least significant bits of the latched address (**A<sub>0</sub>**, **A<sub>1</sub>**) address the two I/O ports and their associated DDR’s.

<u>A<sub>0</sub></u>	<u>A<sub>1</sub></u>	<u>Selection</u>
0	0	Port A
0	1	Port B
1	0	DDR A
1	1	DDR B

## I/O Port Addressing



### Interface to MCS-48™ Microcomputers

## ABSOLUTE MAXIMUM RATINGS\*

Temperature Under Bias	0°C to +70°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin	
With Respect to Ground	-0.3V to +7V
Power Dissipation	1.5W

\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## D.C. CHARACTERISTICS (T<sub>A</sub> = 0°C to 70°C; V<sub>CC</sub> = 5V ± 5%)

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
V <sub>IL</sub>	Input Low Voltage	-0.5	0.8	V	
V <sub>IH</sub>	Input High Voltage	2.0	V <sub>CC</sub> +0.5	V	
V <sub>OL</sub>	Output Low Voltage		0.45	V	I <sub>OL</sub> = 2mA
V <sub>OH</sub>	Output High Voltage	2.4		V	I <sub>OH</sub> = -400μA
I <sub>IL</sub>	Input Leakage		10	μA	V <sub>IN</sub> = V <sub>CC</sub> to 0V
I <sub>LO</sub>	Output Leakage Current		±10	μA	0.45V ≤ V <sub>OUT</sub> ≤ V <sub>CC</sub>
I <sub>CC</sub>	V <sub>CC</sub> Supply Current		180	mA	

## A.C. CHARACTERISTICS (T<sub>A</sub> = 0°C to 70°C; V<sub>CC</sub> = 5V ± 5%)

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
t <sub>CYC</sub>	Clock Cycle Time	320		ns	C <sub>LOAD</sub> = 150 pF (See Figure 3)
T <sub>1</sub>	CLK Pulse Width	80		ns	
T <sub>2</sub>	CLK Pulse Width	120		ns	
t <sub>f</sub> , t <sub>r</sub>	CLK Rise and Fall Time		30	ns	
t <sub>AL</sub>	Address to Latch Set Up Time	50		ns	150 pF Load
t <sub>LA</sub>	Address Hold Time after Latch	80		ns	
t <sub>LC</sub>	Latch to READ/WRITE Control	100		ns	
t <sub>RD</sub>	Valid Data Out Delay from READ Control		150	ns	
t <sub>AD</sub>	Address Stable to Data Out Valid		400	ns	
t <sub>LL</sub>	Latch Enable Width	100		ns	
t <sub>RDF</sub>	Data Bus Float after READ	0	100	ns	
t <sub>CL</sub>	READ/WRITE Control to Latch Enable	20		ns	
t <sub>CC</sub>	READ/WRITE Control Width	250		ns	
t <sub>DW</sub>	Data In to WRITE Set Up Time	150		ns	
t <sub>WD</sub>	Data In Hold Time After WRITE	0		ns	
t <sub>WP</sub>	WRITE to Port Output		400	ns	
t <sub>PR</sub>	Port Input Set Up Time	50		ns	
t <sub>RP</sub>	Port Input Hold Time	50		ns	
t <sub>RYH</sub>	READY HOLD TIME	0	120	ns	
t <sub>ARY</sub>	ADDRESS (CE) to READY		160	ns	
t <sub>RV</sub>	Recovery Time between Controls	300		ns	
t <sub>RDE</sub>	Data Out Delay from READ Control	10		ns	



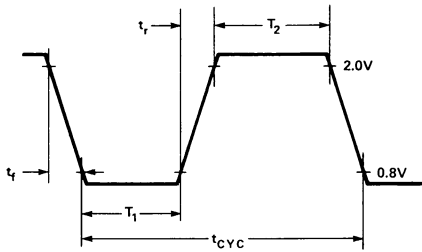


FIGURE 3. CLOCK SPECIFICATION FOR 8355.

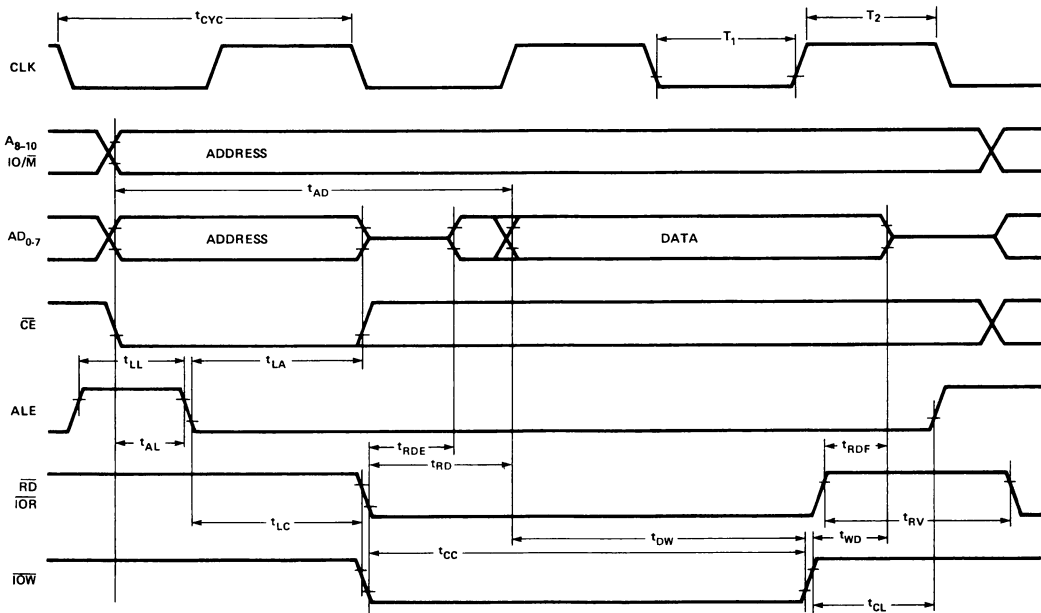


FIGURE 4. ROM READ AND I/O READ AND WRITE.

PRELIMINARY  
 This document is not a final specification. It may be changed without notice.  
 Copyright © 1985 Intel Corporation. All rights reserved.

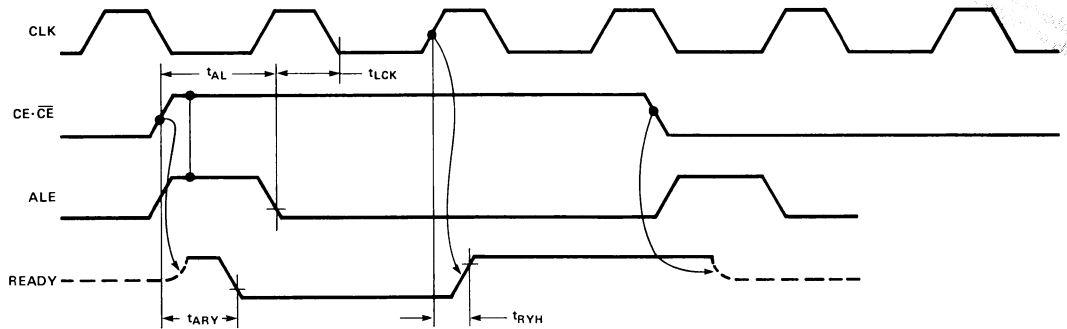
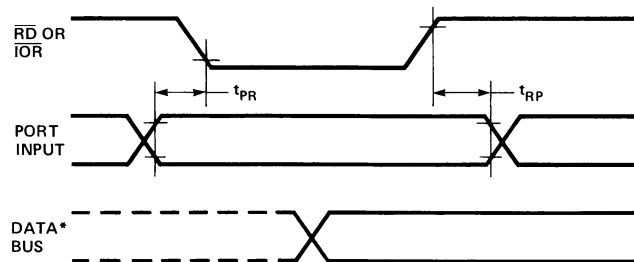
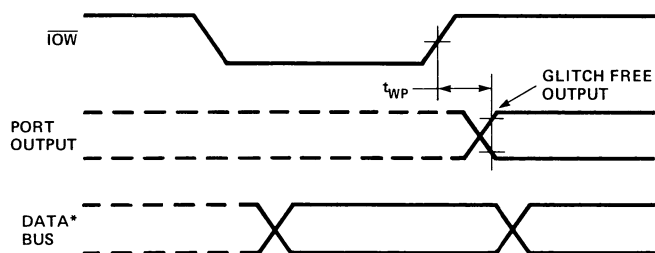


FIGURE 5. WAIT STATE TIMING (READY = 0).

#### A. INPUT MODE



#### B. OUTPUT MODE



\*DATA BUS TIMING IS SHOWN IN FIGURE 3.

FIGURE 6. I/O PORT TIMING.

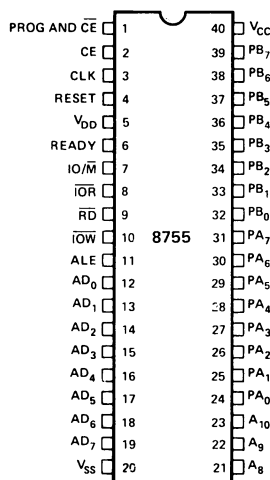
## 8755 EPROM AND I/O EXPANDER

- 2K x 8 EPROM
- 2 Eight Bit I/O Ports
- Internal Address Latch
- I/O Lines Individually Assignable as Input or Output
- Single 5V Supply
- 40 Pin DIP
- Completely Interchangeable With 8355 ROM

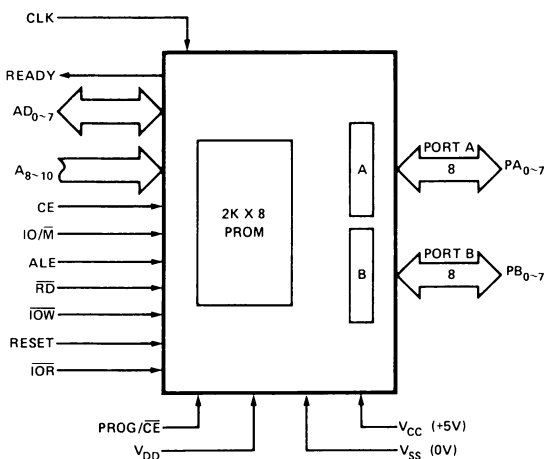
The 8755 is designed to expand both the program memory and I/O capability of the MCS-48™ single component microcomputers (the 8748, 8048 and 8035). This expander increases program memory by 2K words and adds 16 I/O lines to the basic microcomputer without the necessity of any additional components. The completely interchangeable 8755 light erasable EPROM and 8355 mask programmed ROM provide a simple transition from prototype to production. Both versions operate from a single 5V supply and are totally speed compatible with the MCS-48 microcomputers.

The 16 I/O lines are addressed as 2 eight bit I/O ports, yet single lines can be individually designated as input or as output under software control. Outputs are double buffered to prevent any output glitches.

### PIN CONFIGURATION



### BLOCK DIAGRAM



## 8755 FUNCTIONAL PIN DESCRIPTION

Symbol	Function
ALE	When Address Latch Enable is high, AD <sub>0-7</sub> , IO/M, A <sub>8-10</sub> , and CE* (CE* = CE • $\overline{\text{CE}}$ ) enter the address latches. The signals (AD, IO/M, A <sub>8-10</sub> , CE) are latched in at the trailing edge of ALE.
AD <sub>0-7</sub>	Bi-directional Address/Data bus. The lower 8-bits of the PROM or I/O address are applied to the bus lines when ALE is high.  During an I/O cycle, Port A or B are selected based on the latched value of AD <sub>0</sub> . If RD or IOR is low when latched CE* is low, the output buffers present data on the bus.
A <sub>8-10</sub>	These are the high order bits of the PROM address. They do not affect I/O operations.
$\overline{\text{CE}}$ /PROG CE	Both chip enables must be active to permit accessing the PROM. (CE* = CE • $\overline{\text{CE}}$ is low when selected). CE is also used as a programming pin (see section on programming).
IO/M	If the latched IO/M is high when RD is low, the output data comes from an I/O port. If it is low the output data comes from the PROM.
$\overline{\text{RD}}$	If the latched CE* is low when $\overline{\text{RD}}$ goes low, the AD <sub>0-7</sub> output buffers are enabled and output either the selected PROM location or I/O port. When both $\overline{\text{RD}}$ and $\overline{\text{IOR}}$ are high, the AD <sub>0-7</sub> output buffers are tri-stated.
$\overline{\text{IOW}}$	If the latched CE* is low, a low on $\overline{\text{IOW}}$ causes the output port pointed to by the latched value of AD <sub>0</sub> to be written with the data on AD <sub>0-7</sub> . The state of IO/M is ignored.
CLK	The CLK is used to force the READY into its high impedance state after it has been forced low by CE* low and ALE high.
READY	READY is a 3-state output controlled by CE*, ALE and CLK. READY is forced low by CE* during the time ALE is high, and remains low until the rising edge of the next CLK (see Figure 2).
PA <sub>0-7</sub>	These are general purpose I/O pins. Their input/output direction is determined by the contents of Data Direction Register (DDR). Port A is selected for write operations by CE* and $\overline{\text{IOW}}$ low and a 0 previously latched from AD <sub>0</sub> .  Read operation is selected by either $\overline{\text{IOR}}$ low or IO/M high and $\overline{\text{RD}}$ low, and the latched CE* low and AD <sub>0</sub> low.

PB <sub>0-7</sub>	This general purpose I/O port is identical to Port A except that it is selected by a 1 latched from AD <sub>0</sub> .
RESET	In normal operation, an input high on RESET causes all pins in Ports A and B to assume input mode (clear DDR register).
$\overline{\text{IOR}}$	When CE* is low, a low on $\overline{\text{IOR}}$ will output the selected I/O port onto the AD bus. $\overline{\text{IOR}}$ low performs the same function as the combination of IO/M high and RD low. When $\overline{\text{IOR}}$ is not used in a system, $\overline{\text{IOR}}$ should be tied to VCC ("1").
V <sub>CC</sub>	+5 volt supply.
V <sub>SS</sub>	0 volt supply.
V <sub>DD</sub>	V <sub>DD</sub> is a programming voltage, and it is normally grounded.  For programming, a high voltage is supplied with V <sub>DD</sub> , = 25V, typical.

### PROM Section

The PROM section of the chip is addressed by the 11-bit address and CE. The address and CE are latched into the address latches on the falling edge of ALE. If the latched CE\* is low and IO/M is low when RD goes low, the eight PROM bits addressed by the latched address are put out through AD<sub>0-7</sub> output buffers.

### I/O Section

The I/O section of the chip is addressed by the latched value of AD<sub>0-1</sub> and CE\*. Two 8-bit Data Direction Registers determine the input/output status of each pin in the corresponding port. A 0 specifies an input mode, and a 1 specifies an output mode. The table summarizes port and DDR designation. Contents of the DDR's cannot be read.

AD <sub>1</sub>	AD <sub>0</sub>	Selection
0	0	Port A
0	1	Port B
1	0	Port A Data Direction Register (DDR A)
1	1	Port B Data Direction Register (DDR B)

When  $\overline{\text{IOW}}$  goes low and CE\* is low, the data on the AD<sub>0-7</sub> is written into I/O port selected by the latched value of AD<sub>0-1</sub>. During this operation all I/O bits of the selected port are affected, regardless of their I/O mode and the state of IO/M. The actual output level does not change until  $\overline{\text{IOW}}$  returns high. (glitch free output).

A port can be read out when the latched CE\* is low and either  $\overline{\text{RD}}$  goes low with IO/M high, or  $\overline{\text{IOR}}$  goes low. Both input and output mode bits of a selected port will appear on lines AD<sub>0-7</sub>.

### Programming

The word to be programmed is selected by latching the proper 11-bit address and CE\* into the PROM with ALE. Data presented on the AD<sub>0-7</sub> lines is programmed into that word by a high level TTL pulse on the  $\overline{\text{CE}}$ /PROG pin. The pulse should typically be 50 msec long with 26V on V<sub>DD</sub>, or the PROG pin can remain high and V<sub>DD</sub> can be pulsed for 100 ms.

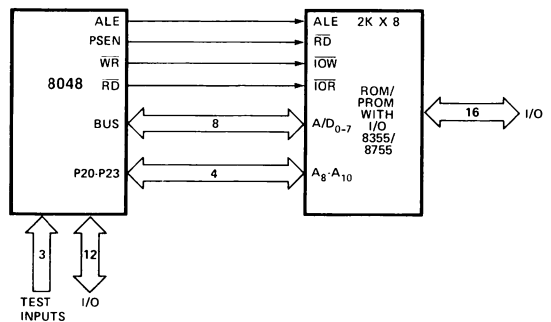
## FUNCTIONAL DESCRIPTION

**Program Memory** — The 8755 contains an 8-bit address latch which allows it to interface directly to MCS-48 Microcomputers without additional hardware. Program memory is accessed by applying 11 bits of address to the  $A_0 - A_{10}$  inputs and a low level on the  $\overline{IO/M}$  and  $\overline{CE}$  inputs then latching these inputs with ALE. The  $\overline{CE}$  input serves to select one of several possible 8755s in a system and the  $\overline{IO/M}$  signal indicates that a subsequent read operation will be from program memory. While ALE is high the  $A_0 - A_{10}$ ,  $\overline{IO/M}$ , and  $\overline{CE}$  inputs are allowed into the 8755 and when ALE is brought low, these inputs are latched. If the latched conditions indicate that a program memory fetch is to occur, a low level on  $\overline{RD}$  will cause the data to be outputted on the data bus.

**I/O Ports** — The I/O lines are organized as two 8-bit static ports which can be read or written using the  $\overline{IOR}$  and  $\overline{IOW}$  control lines. Associated with each port is an 8-bit Data Direction Register (DDR) which serves to define each of the 8 lines of the port as either an input or an output. A "1" bit in the DDR sets the corresponding port bit to the output mode while a "0" designates the input mode. The two least significant bits of the latched address ( $A_0$ ,  $A_1$ ) address the two I/O ports and their associated DDR's.

$A_0$	$A_1$	Selection
0	0	Port A
0	1	Port B
1	0	DDR A
1	1	DDR B

### I/O Port Addressing



### Interface to MCS-48™ Microcomputers

**ABSOLUTE MAXIMUM RATINGS\***

Temperature Under Bias	-10°C to +70°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin	
With Respect to Ground	-0.5V to +7V
Power Dissipation	1.5W

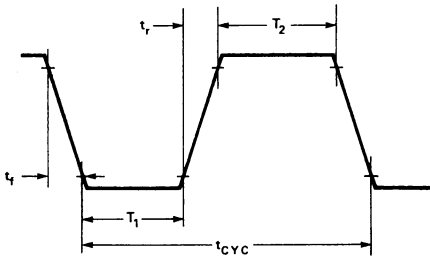
\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**D.C. CHARACTERISTICS** ( $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ;  $V_{CC} = 5V \pm 5\%$ )

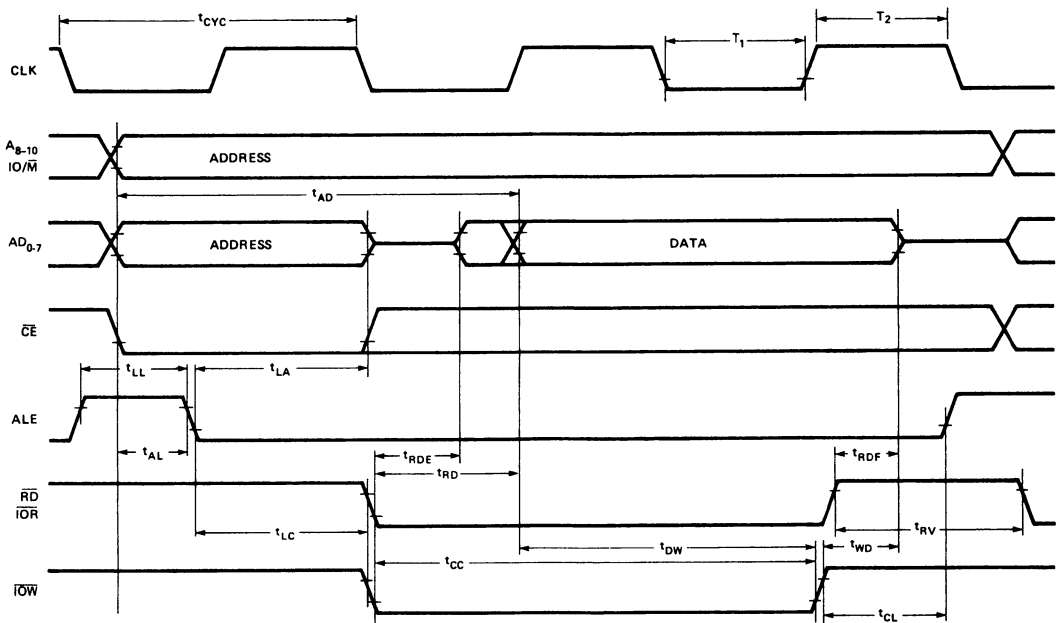
SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$V_{IL}$	Input Low Voltage	-0.5	0.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC}+0.5$	V	
$V_{OL}$	Output Low Voltage		0.45	V	$I_{OL} = 2\text{mA}$
$V_{OH}$	Output High Voltage	2.4		V	$I_{OH} = -400\mu\text{A}$
$I_{IL}$	Input Leakage		10	$\mu\text{A}$	$V_{IN} = V_{CC}$ to 0V
$I_{LO}$	Output Leakage Current		$\pm 10$	$\mu\text{A}$	$0.45V \leq V_{OUT} \leq V_{CC}$
$I_{CC}$	$V_{CC}$ Supply Current		180	mA	

**A.C. CHARACTERISTICS** ( $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ;  $V_{CC} = 5V \pm 5\%$ )

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$t_{CYC}$	Clock Cycle Time	320		ns	$C_{LOAD} = 150\text{ pF}$ (See Figure 3)
$T_1$	CLK Pulse Width	80		ns	
$T_2$	CLK Pulse Width	120		ns	
$t_r, t_f$	CLK Rise and Fall Time		30	ns	
$t_{AL}$	Address to Latch Set Up Time	50		ns	150 pF Load
$t_{LA}$	Address Hold Time after Latch	80		ns	
$t_{LC}$	Latch to READ/WRITE Control	100		ns	
$t_{RD}$	Valid Data Out Delay from READ Control		150	ns	
$t_{AD}$	Address Stable to Data Out Valid		400	ns	
$t_{LL}$	Latch Enable Width	100		ns	
$t_{RDF}$	Data Bus Float after READ	0	100	ns	
$t_{CL}$	READ/WRITE Control to Latch Enable	20		ns	
$t_{CC}$	READ/WRITE Control Width	250		ns	
$t_{DW}$	Data In to WRITE Set Up Time	150		ns	
$t_{WD}$	Data In Hold Time After WRITE	0		ns	
$t_{WP}$	WRITE to Port Output		400	ns	
$t_{PR}$	Port Input Set Up Time	50		ns	
$t_{RP}$	Port Input Hold Time	50		ns	
$t_{RYH}$	READY HOLD TIME	0	120	ns	
$t_{ARY}$	ADDRESS (CE) to READY		160	ns	
$t_{RV}$	Recovery Time between Controls	300		ns	
$t_{RDE}$	Data Out Delay from READ Control	10		ns	

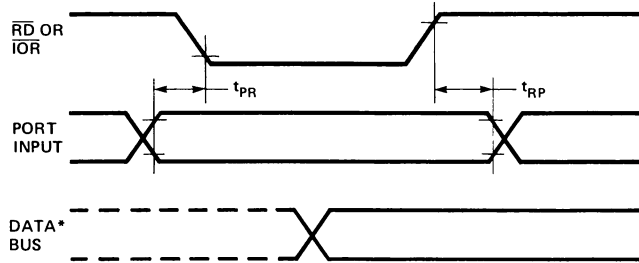


**FIGURE 3. CLOCK SPECIFICATION FOR 8755**

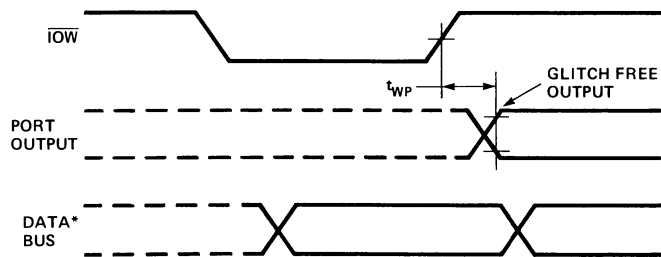


**FIGURE 4. PROM READ AND I/O WRITE TIMING.**

## A. INPUT MODE



## B. OUTPUT MODE



\*DATA BUS TIMING IS SHOWN IN FIGURE 4.

FIGURE 5. I/O PORT TIMING.

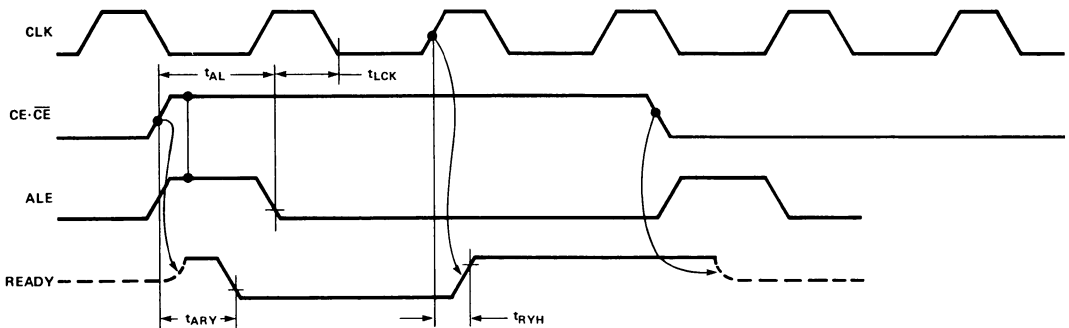


FIGURE 6. WAIT STATE TIMING (READY = 0).



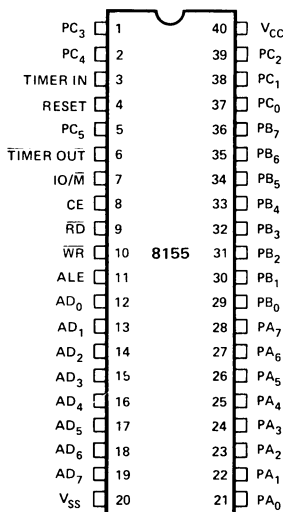
## 8155 RAM AND I/O EXPANDER

- 256 x 8 Static RAM
- 2 Programmable 8-Bit I/O Ports
- 1 Programmable 6-Bit I/O Port
- Internal Address Latch
- Single 5V Supply
- 40 Pin Dual-In-Line Package
- Programmable 14-Bit Timer/Counter

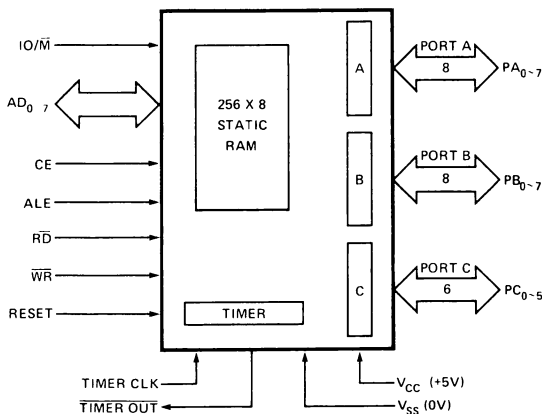
The 8155 is designed to expand the data memory, I/O, and timer capability of the MCS-48™ single component microcomputers (the 8748, 8048, and 8035). This expander increases data memory by 256 words, adds 22 I/O lines, and adds a 14 bit timer/counter to the basic microcomputer without the necessity of any additional components.

The data memory is a 256 x 8 static RAM which is speed compatible with all MCS-48 components. The I/O consists of two eight-bit ports which can be programmed for either input or output with or without associated handshaking signals and processor interrupt requests. An additional 6-bit port functions as an input port, as an output port, or as the source of strobes for the two eight-bit ports in the handshake mode. The 14 bit programmable timer/counter whose input clock and terminal count output are available to the user externally is programmable for several modes of operation.

### PIN CONFIGURATION



### BLOCK DIAGRAM



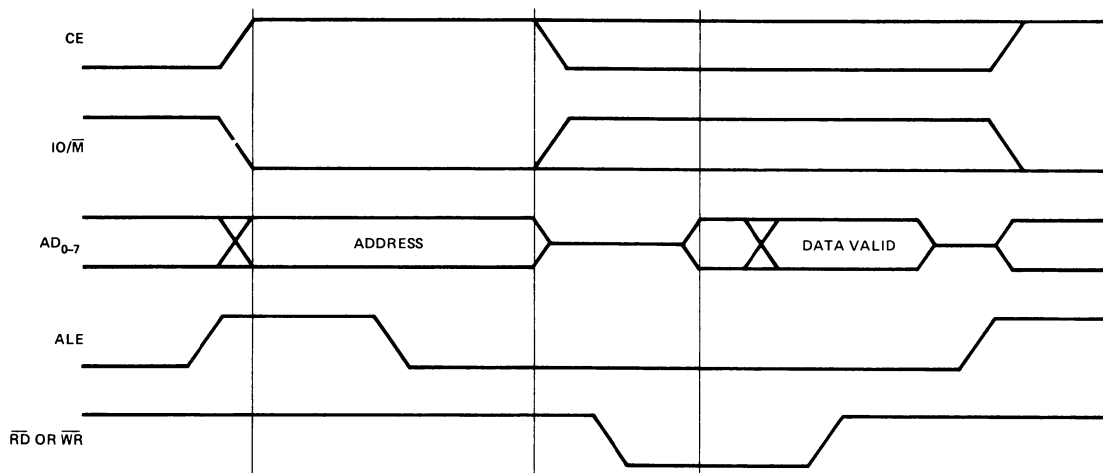
## OPERATIONAL DESCRIPTION

The 8155 includes the following operational features:

- 2K Bit Static RAM organized as 256 x 8
- Two 8-bit I/O ports (PA & PB) and one 6-bit I/O port (PC)
- 14-bit binary down counter

The I/O portion contains four registers (Command/Status, PA<sub>0-7</sub>, PB<sub>0-7</sub>, PC<sub>0-5</sub>). The IO/M (IO/Memory Select) pin selects the I/O or the memory (RAM) portion. Detailed descriptions of memory, I/O ports and timer functions will follow.

The 8-bit address on the AD lines, CE, and IO/M are all latched on chip at the falling edge of ALE. Therefore the ALE signal should be activated (high) before the transition of CE and IO/M signal, as shown in Figure 1. A low on the IO/M must be provided to select the memory section.



NOTE: FOR DETAILED TIMING DIAGRAM INFORMATION, SEE FIGURE 7 AND A.C. CHARACTERISTICS.

**FIGURE 1. MEMORY READ/WRITE CYCLE.**

## PROGRAMMING OF THE COMMAND/STATUS REGISTER

The command register consists of eight latches one for each bit. Four bits (0-3) define the mode of the ports, two bits (4-5) enable or disable the interrupt from port C when it acts as control port, and the last two bits (6-7) are for the timer.

The C/S register contents can be altered at any time by using the I/O address XXXXX000 during a WRITE operation. The meaning of each bit of the command byte is defined as follows:

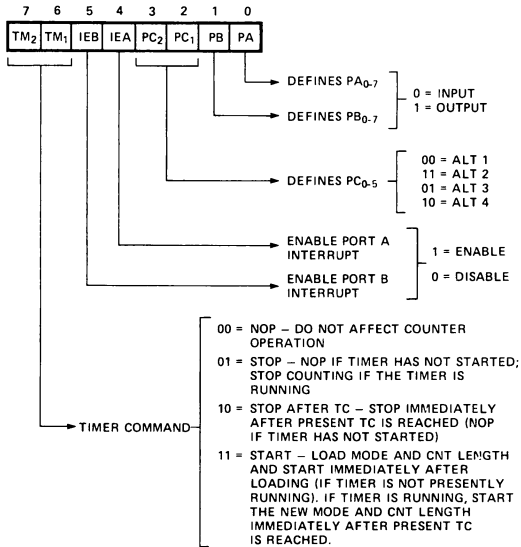


FIGURE 2. COMMAND/STATUS REGISTER BIT ASSIGNMENT.

## READING THE COMMAND/STATUS REGISTER

The status register consists of seven latches one for each bit; six (0-5) for the status of the ports and one (6) for the status of the timer. The timer bit consists of a flip flop and a latch.

The status of the timer and the I/O section can be polled by reading the C/S Register (Address XXXXX000). Status word format is shown below:

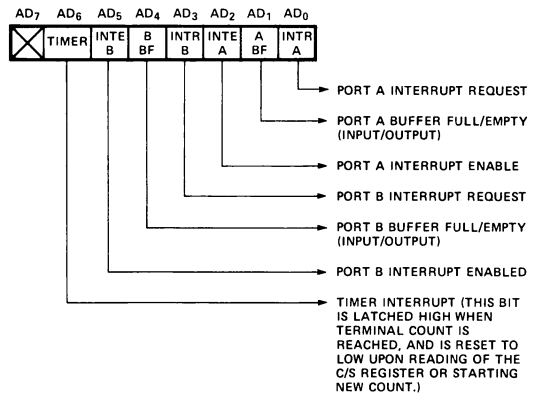


FIGURE 3. COMMAND/STATUS REGISTER STATUS WORD FORMAT.

## INPUT/OUTPUT SECTION

The I/O section of the 8155 consists of four registers as described below:

- **Command/Status Register (C/S)** — This register is assigned the address XXXXX000. The C/S address serves the dual purpose.

When the C/S register is selected during WRITE operation, a command is written into the command register. The contents of this register are *not* accessible through the pins.

When the C/S (XXXXX000) is selected during a READ operation, the status information of the I/O ports and the timer become available on the AD<sub>0-7</sub> lines.

- **PA Register** — This register can be programmed to be either input or output ports depending on the status of the contents of the C/S Register. Also depending on the command, this port can operate in either the basic mode or the strobed mode (See timing diagram). The I/O pins assigned in relation to this register are PA<sub>0-7</sub>. The address of this register is XXXXX001.

- **PB Register** — This register functions the same as PA Register. The I/O pins assigned are PB<sub>0-7</sub>. The address of this register is XXXXX010.

- **PC Register** — This register has the address XXXXX011 and contains only 6-bits. The 6-bits can be programmed to be either input ports, output ports or as control signals for PA and PB by properly programming the AD<sub>2</sub> and AD<sub>3</sub> bits of the C/S register.

When PC<sub>0-5</sub> is used as a control port, 3-bits are assigned for Port A and 3 for Port B. The first bit is an interrupt that the 8155 sends out. The second is an output signal indicating whether the buffer is full or empty, and the third is an input pin to accept a strobe for the strobed input mode. See Table 1.

**TABLE 1. TABLE OF PORT CONTROL ASSIGNMENT.**

Pin	ALT 1	ALT 2	ALT 3	ALT 4
PC0	Input Port	Output Port	A INTR (Port A Interrupt)	A INTR (Port A Interrupt)
PC1	Input Port	Output Port	A BF (Port A Buffer Full)	A BF (Port A Buffer Full)
PC2	Input Port	Output Port	A $\overline{STB}$ (Port A Strobe)	A $\overline{STB}$ (Port A Strobe)
PC3	Input Port	Output Port	Output Port	B INTR (Port B Interrupt)
PC4	Input Port	Output Port	Output Port	B BF (Port B Buffer Full)
PC5	Input Port	Output Port	Output Port	B $\overline{STB}$ (Port B Strobe)

The set and reset of INTR and BF with respect to  $\overline{STB}$ ,  $\overline{WR}$  and  $\overline{RD}$  timing is shown in Figure 9.

In the summary, the registers' assignments are:

Address	Pinouts	Functions	No. of Bits
XXXXX000	Internal	Command/Status Register	8
XXXXX001	PA0-7	General Purpose I/O Port	8
XXXXX010	PB0-7	General Purpose I/O Port	8
XXXXX011	PC0-5	General Purpose I/O Port or Control Lines	6

When the I/O ports are programmed to be output ports, the contents of the output ports can still be read by a READ operation when appropriately addressed.

When the 'C' port is programmed to either ALT3 or ALT4, the control signals for PA and PB are initialized as follows:

CONTROL	INPUT MODE	OUTPUT MODE
BF	Low	Low
INTR	Low	High
$\overline{STROB}$	Input Control	Input Control

## TIMER SECTION

The timer is a 14-bit counter that counts the 'timer input' pulses and provides either a square wave or pulse when terminal count (TC) is reached.

The timer has the I/O address XXXXX100 for the low order byte of the register and the I/O address XXXXX101 for the high order byte of the register.

The timer addresses serve a dual purpose. During WRITE operation, a COUNT LENGTH REGISTER (CLR) with a count length (bits 0-13) and a timer mode (bits 14-15) are loaded. During READ operation the contents of the counter (the present count) and the mode bits are read.

To be sure that the right content of the counter is read, it is preferable to stop counting, read it, and then load it again and continue counting.

To program the timer, the COUNT LENGTH REG is loaded first, one byte at a time, by selecting the timer addresses. Bits 0-13 will specify the length of the next count and bits 14-15 will specify the timer output mode.

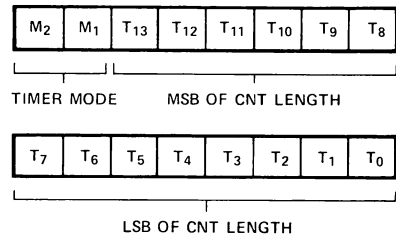
There are four modes to choose from:

0. Puts out low during second half of count.
1. Square wave
2. Single pulse upon TC being reached
3. Repetitive single pulse everytime TC is reached and automatic reload of counter upon TC being reached, until instructed to stop by a new command loaded into C/S.

Bits 6-7 of Command/Status Register Contents are used to start and stop the counter. There are four commands to choose from:

*Note: See the further description on Command/Status Register.*

C/S7	C/S6	
0	0	NOP — Do not affect counter operation.
0	1	STOP — NOP if timer has not started; stop counting if the timer is running.
1	0	STOP AFTER TC — Stop immediately after present TC is reached (NOP if timer has not started)
1	1	START — Load mode and CNT length and start immediately after loading (if timer is not presently running). If timer is running, start the new mode and CNT length immediately after present TC is reached.

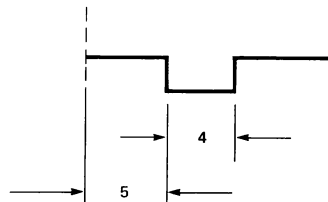


**FIGURE 4. TIMER FORMAT**

M2 M1 defines the timer mode as follows:

M2	M1	
0	0	Puts out low during second half of count.
0	1	Square wave, i.e., the period of the square wave equals the count length programmed with automatic reload at terminal count.
1	0	Single pulse upon TC being reached.
1	1	Automatic reload, i.e., single pulse everytime TC is reached.

*Note: In case of an asymmetric count, i.e. 9, larger half of the count will be high, the larger count will stay active as shown in Figure 5.*



*Note: 5 and 4 refer to the number of clock cycles in that time period.*

**FIGURE 5. ASYMMETRIC COUNT.**

## ABSOLUTE MAXIMUM RATINGS\*

Temperature Under Bias .....	0°C to +70°C
Storage Temperature .....	-65°C to +150°C
Voltage on Any Pin	
With Respect to Ground .....	-0.3V to +7V
Power Dissipation .....	1.5W

\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## D.C. CHARACTERISTICS (T<sub>A</sub> = 0°C to 70°C; V<sub>CC</sub> = 5V ± 5%)

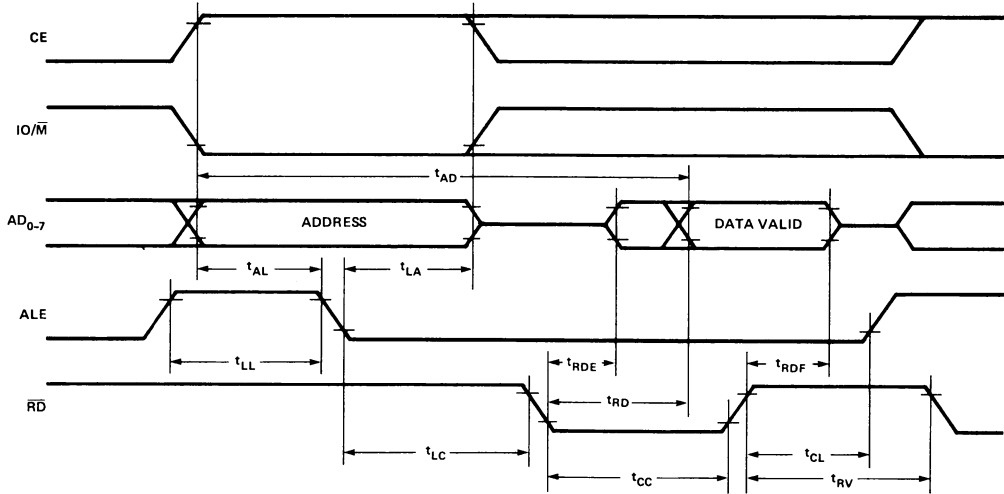
SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
V <sub>IL</sub>	Input Low Voltage	-0.5	0.8	V	
V <sub>IH</sub>	Input High Voltage	2.0	V <sub>CC</sub> +0.5	V	
V <sub>OL</sub>	Output Low Voltage		0.45	V	I <sub>OL</sub> = 2mA
V <sub>OH</sub>	Output High Voltage	2.4		V	I <sub>OH</sub> = -400μA
I <sub>IL</sub>	Input Leakage		10	μA	V <sub>IN</sub> = V <sub>CC</sub> to 0V
I <sub>LO</sub>	Output Leakage Current		±10	μA	0.45V ≤ V <sub>OUT</sub> ≤ V <sub>CC</sub>
I <sub>CC</sub>	V <sub>CC</sub> Supply Current		180	mA	

**A.C. CHARACTERISTICS** ( $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ;  $V_{CC} = 5V \pm 5\%$ )

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$t_{AL}$	Address to Latch Set Up Time	50		ns	150 pF Load
$t_{LA}$	Address Hold Time after Latch	80		ns	
$t_{LC}$	Latch to READ/WRITE Control	100		ns	
$t_{RD}$	Valid Data Out Delay from READ Control		150	ns	
$t_{AD}$	Address Stable to Data Out Valid		400	ns	
$t_{LL}$	Latch Enable Width	100		ns	
$t_{RDF}$	Data Bus Float After READ	0	100	ns	
$t_{CL}$	READ/WRITE Control to Latch Enable	20		ns	
$t_{CC}$	READ/WRITE Control Width	250		ns	
$t_{DW}$	Data In to WRITE Set Up Time	150		ns	
$t_{WD}$	Data In Hold Time After WRITE	0		ns	
$t_{RV}$	Recovery Time Between Controls	300		ns	
$t_{WP}$	WRITE to Port Output		400	ns	
$t_{PR}$	Port Input Setup Time	50		ns	
$t_{RP}$	Port Input Hold Time	50		ns	
$t_{SBF}$	Strobe to Buffer Full		400	ns	
$t_{SS}$	Strobe Width	200		ns	
$t_{RBE}$	READ to Buffer Empty		400	ns	
$t_{SI}$	Strobe to INTR On		400	ns	
$t_{RDI}$	READ to INTR Off		400	ns	
$t_{PSS}$	Port Setup Time to Strobe Strobe	50		ns	
$t_{PHS}$	Port Hold Time After Strobe	100		ns	
$t_{SBE}$	Strobe to Buffer Empty		400	ns	
$t_{WBF}$	WRITE to Buffer Full		400	ns	
$t_{WI}$	WRITE to INTR Off		400	ns	
$t_{TL}$	TIMER-IN to $\overline{\text{TIMER-OUT}}$ Low	400		ns	
$t_{TH}$	TIMER-IN to $\overline{\text{TIMER-OUT}}$ High	400		ns	
$t_{RDE}$	Data Bus Enable from READ Control	10		ns	

Note: For Timer Input Specification, see Figure 10.

### A. READ CYCLE



### B. WRITE CYCLE

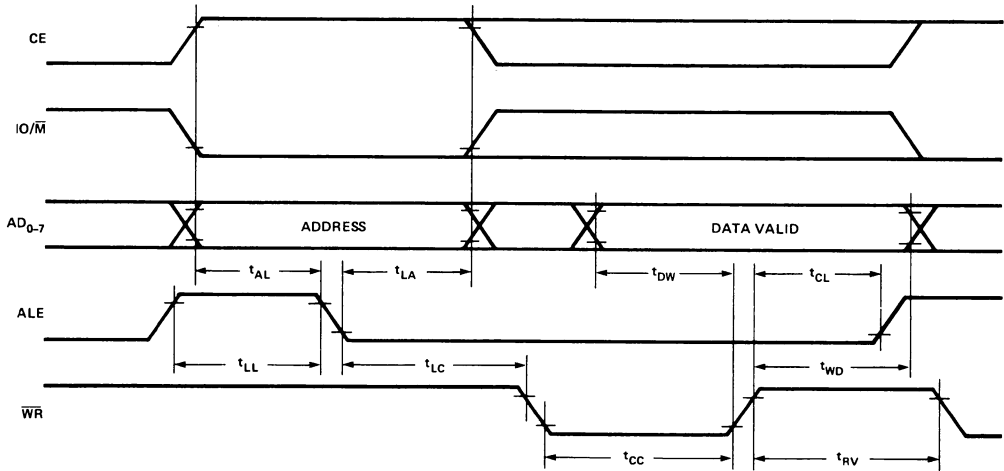
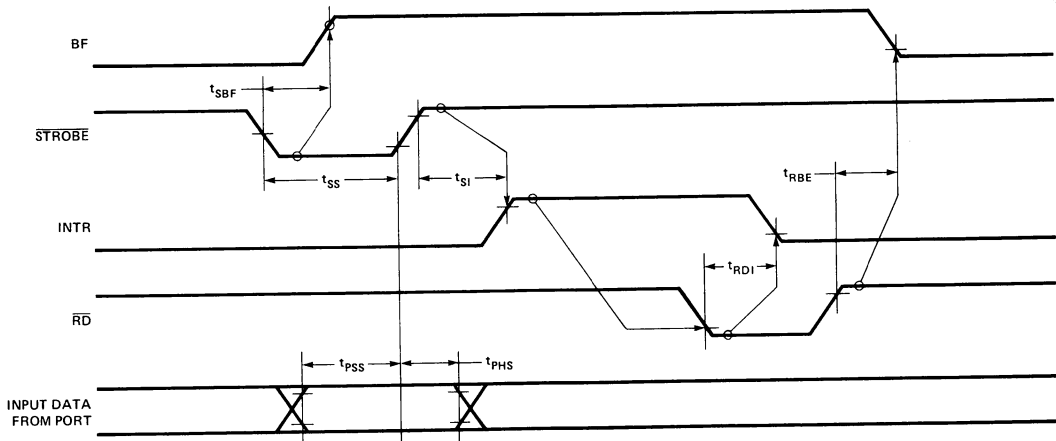


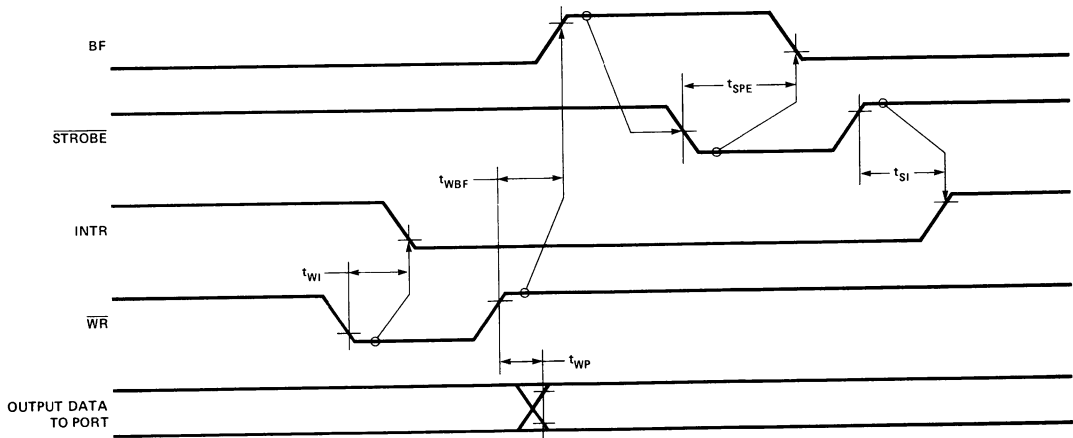
FIGURE 7. 8155 READ/WRITE TIMING DIAGRAM.



### A. STROBED INPUT MODE



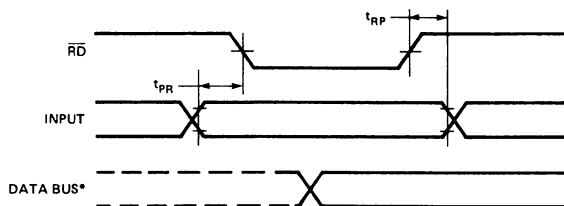
### B. STROBED OUTPUT MODE



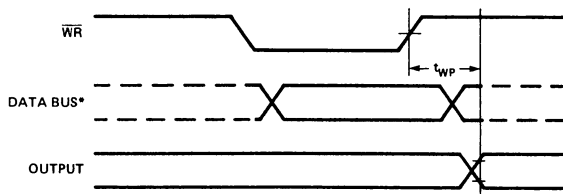
**FIGURE 8. BASIC I/O TIMING.**

**PRELIMINARY**  
 Not for use in a final specification. Some parameters may be subject to change.

### A. BASIC INPUT MODE



### B. BASIC OUTPUT MODE



\*DATA BUS TIMING IS SHOWN IN FIGURE 7.

FIGURE 9. STROBED I/O TIMING WAVEFORM.

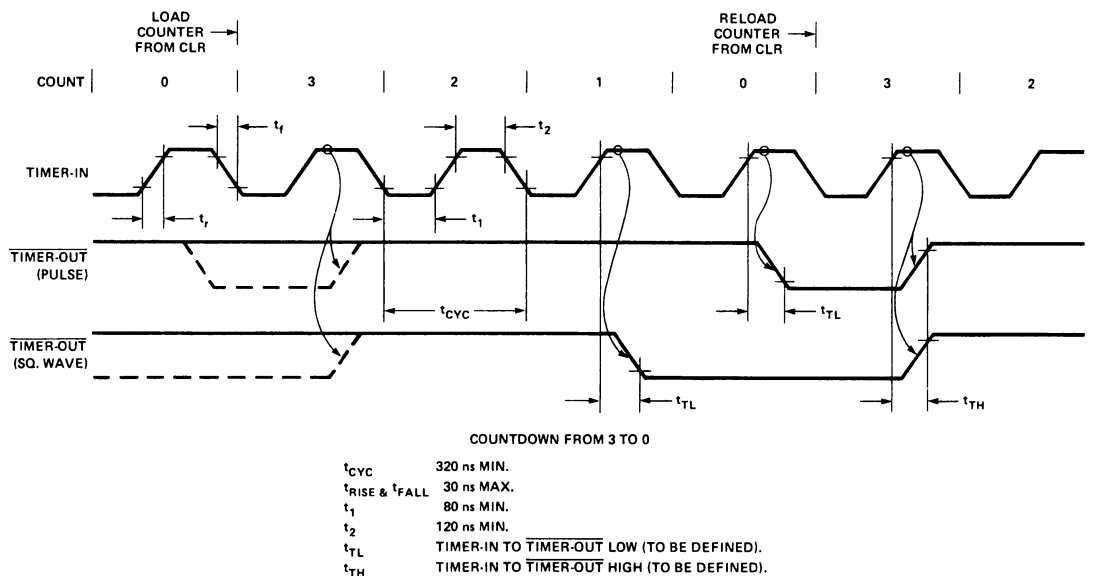


FIGURE 10. TIMER OUTPUT WAVEFORM.

# 8243 MCS-48™ INPUT/OUTPUT EXPANDER

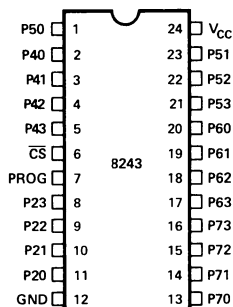
- Low Cost
- Simple Interface to MCS-48™ Micro-computers
- Four 4-Bit I/O Ports
- AND and OR Directly to Ports
- 24 Pin DIP
- Single 5V Supply
- High Output Drive
- Direct Extension of Resident 8048 I/O Ports

The 8243 is an input/output expander designed specifically to provide a low cost means of I/O expansion for the MCS-48 family of single-chip microcomputers. Fabricated in 5 volts NMOS, the 8243 combines low cost, single supply voltage and high drive current capability.

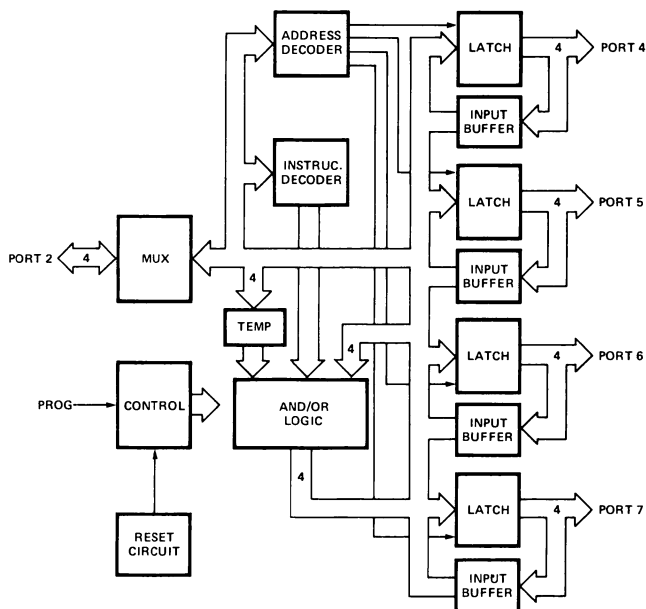
The 8243 consists of four 4-bit bi-directional static I/O ports and one 4-bit port which serves as an interface to the MCS-48 microcomputers. The 4-bit interface requires that only four (4) I/O lines of the 8048 be used for I/O expansion and also allows multiple 8243's to be added to the same bus.

The I/O ports of the 8243 serve as a direct extension of the resident I/O facilities of the MCS-48 microcomputers and are accessed by their own MOV, ANL, and ORL instructions.

**PIN CONFIGURATION**



**BLOCK DIAGRAM**



## PIN DESCRIPTION

Symbol	Pin No.	Function
PROG	7	Clock Input. A high to low transition on PROG signifies that address and control are available on P20-P23, and a low to high transition signifies that data is available on P20-23.
$\overline{CS}$	6	Chip Select Input. A high on CS inhibits any change of output or internal status.
P20-P23	11-8	Four (4) bit bi-directional port contains the address and control bits on a high to low transition of PROG. During a low to high transition contains the data for a selected output port if a write operation, or the data from a selected port before the low to high transition if a read operation.
GND	12	0 volt supply.
P40-P43	2-5	Four (4) bit bi-directional I/O ports. May be programmed to be input (during read), low impedance latched output (after write) or a tri-state (after read). Data on pins P20-23 may be directly written, ANDed or ORed with previous data.
P50-P53	1,23-21	
P60-P63	20-17	
P70-P73	13-16	
$V_{CC}$	24	+5 volt supply.

## FUNCTIONAL DESCRIPTION

### General Operation

The 8243 contains four 4-bit I/O ports which serve as an extension of the on-chip I/O and are addressed as ports 4-7. The following operations may be performed on these ports:

- Transfer Accumulator to Port.
- Transfer Port to Accumulator.
- AND Accumulator to Port.
- OR Accumulator to Port.

All communication between the 8048 and the 8243 occurs over Port 2 (P20-P23) with timing provided by an output pulse on the PROG pin of the processor. Each transfer consists of two 4-bit nibbles:

The first containing the "op code" and port address and the second containing the actual 4-bits of data.

A high to low transition of the PROG line indicates that address is present while a low to high transition indicates the presence of data. Additional 8243's may be added to the 4-bit bus and chip selected using additional output lines from the 8048/8748/8035.

### Power On Initialization

Initial application of power to the device forces input/output ports 4, 5, 6, and 7 to the tri-state and port 2 to the input mode. The PROG pin may be either high or low when power is applied. The first high to low transition of PROG causes device to exit power on mode. The power on sequence is initiated if  $V_{CC}$  drops below 1V.

P21	P20	Address Code	P23	P22	Instruction Code
0	0	Port 4	0	0	Read
0	1	Port 5	0	1	Write
1	0	Port 6	1	0	ORLD
1	1	Port 7	1	1	ANLD

### Write Modes

The device has three write modes. MOVD  $P_i$ , A directly writes new data into the selected port and old data is lost. ORLD  $P_i$ , A takes new data, OR's it with the old data and then writes it to the port. ANLD  $P_i$ , A takes new data AND's it with the old data and then writes it to the port. Operation code and port address are latched from the input port 2 on the high to low transition of the PROG pin. On the low to high transition of PROG data on port 2 is transferred to the logic block of the specified output port.

After the logic manipulation is performed, the data is latched and outputted. The old data remains latched until new valid outputs are entered.

### Read Mode

The device has one read mode. The operation code and port address are latched from the input port 2 on the high to low transition of the PROG pin. As soon as the read operation and port address are decoded, the appropriate outputs are tri-stated, and the input buffers switched on. The read operation is terminated by a low to high transition of the PROG pin. The port (4, 5, 6 or 7) that was selected is switched to the tri-stated mode while port 2 is returned to the input mode.

Normally, a port will be in an output (write mode) or input (read mode). If modes are changed during operation, the first read following a write should be ignored; all following reads are valid. This is to allow the external driver on the port to settle after the first read instruction removes the low impedance drive from the 8243 output.

**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias. . . . . 0°C to 70°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage on Any Pin  
     With Respect to Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

*\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

**D.C. AND OPERATING CHARACTERISTICS**

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$

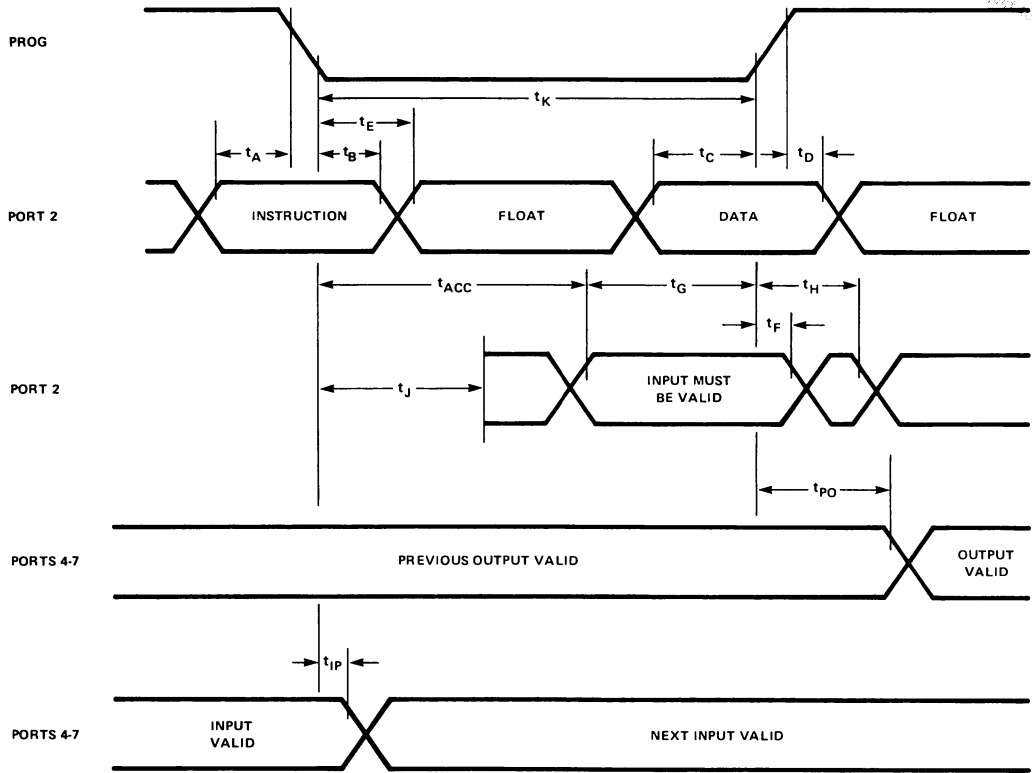
SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$V_{IL}$	Input Low Voltage	-0.5	0.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC}+0.5$	V	
$V_{OL1}$	Output Low Voltage Ports 4-7		0.45	V	$I_{OL} = 10\text{ mA}$
$V_{OL2}$	Output Low Voltage Ports 4-7		TBD	V	$I_{OL} = 20\text{ mA}$
$V_{OH}$	Output High Voltage		2.4	V	$I_{OH} = -400\mu\text{A}$
$I_{IL}$	Input Leakage		10	$\mu\text{A}$	$V_{in} = V_{CC}$ to 0V
$I_{OL2}$	Output Current Port 2	2.5	8.5	mA	$V_{OL} = 1.5\text{V}$
$V_{OL3}$	Output Low Voltage Port 2		.45	V	$I_{OL} = 0.8\text{mA}$
$I_{CC}$	$V_{CC}$ Supply Current		40	mA	
$I_{OH}$	Output Current Port 2		TBD		$V_{OH} = 2.4\text{V}$
$I_{VSS}$	$I_{CC}$ Plus Sum of all $I_{OL}$ from 16 Outputs		120	mA	

**A.C. CHARACTERISTICS**

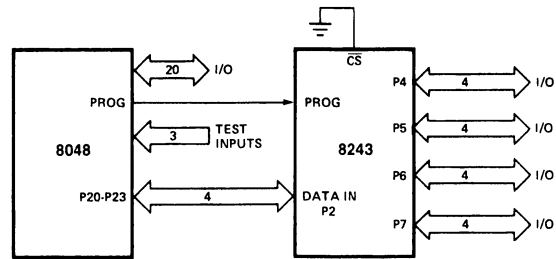
$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$t_A$	Code Valid Before PROG	70		ns	80 pF Load
$t_B$	Code Valid After PROG	60		ns	20 pF Load
$t_C$	Data Valid Before PROG	200		ns	80 pF Load
$t_D$	Data Valid After PROG	20		ns	20 pF Load
$t_E$	Float After PROG		700	ns	
$t_J$	Enabled After PROG	200		ns	
$t_G$	Data Valid Before PROG	300		ns	80 pF Load
$t_F$	Data Valid After PROG	20		ns	20 pF Load
$t_H$	Floating After PROG		100	ns	20 pF Load
$t_K$	PROG Negative Pulse Width	900		ns	
$t_{CP}$	CS Valid Before PROG	TBD			
$t_{PC}$	CS Valid Before PROG	TBD			
$t_{PO}$	Ports 4-7 Valid After PROG		TBD		100 pF Load
$t_{IP}$	Ports 4-7 Valid Before/After PROG	TBD			
$t_{ACC}$	Port 2 Valid After PROG	400		ns	80 pF Load

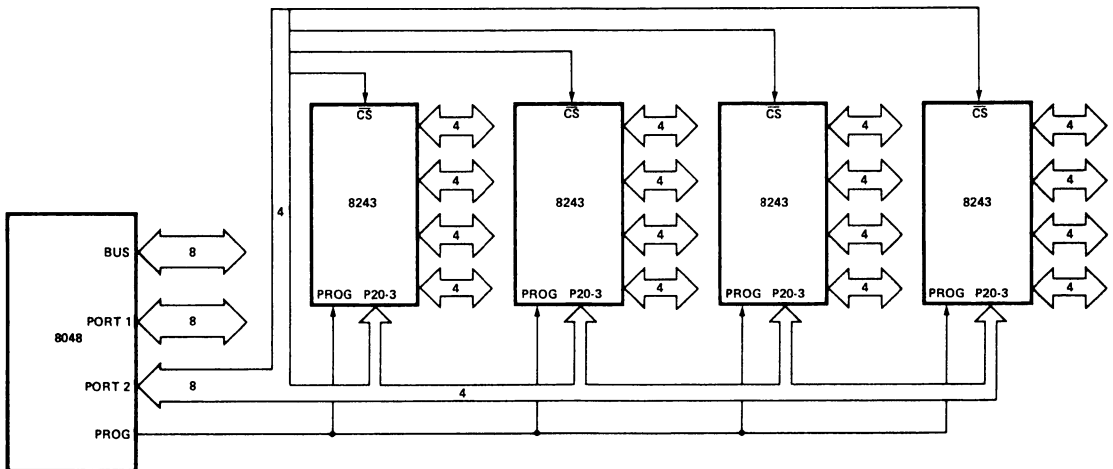
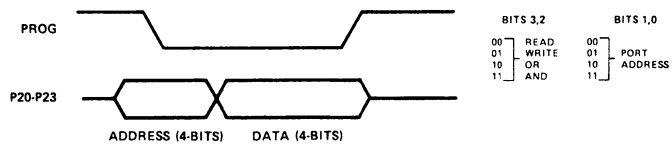
## WAVEFORMS



## EXPANDER INTERFACE



## OUTPUT EXPANDER TIMING

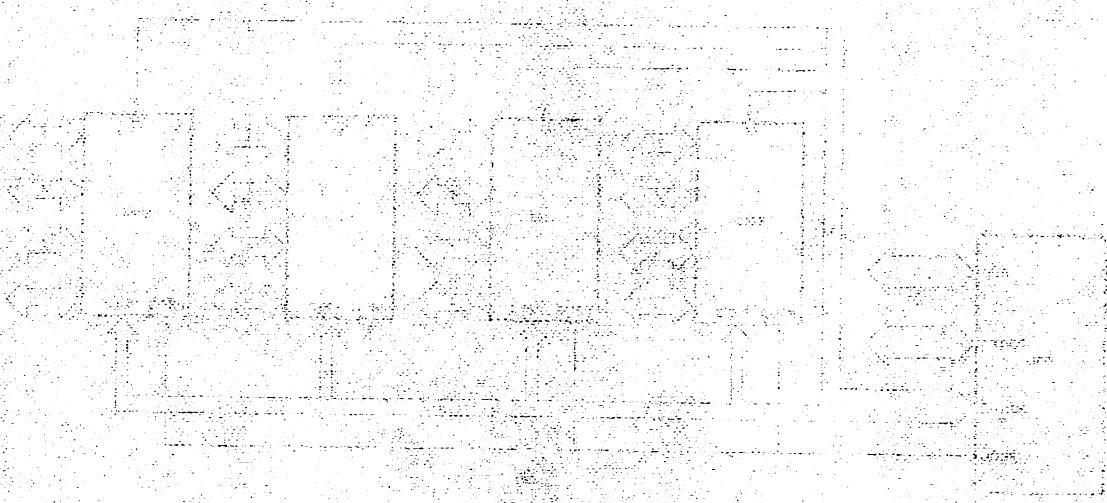


## USING MULTIPLE 8243's

# 10-1



## 10-2



## 10-3



## Chapter 7

# COMPATIBLE MCS-80™ COMPONENTS



## COMPATIBLE MCS-80™ COMPONENTS

8308	8192 Bit Static MOS ROM .....	7-1
8316A	16,384 Bit Static MOS ROM .....	7-5
8708	8192 1K x 8 EPROM .....	7-11
8101A-4	1024 Bit Static MOS RAM With Separate I/O .....	7-15
8111A-4	1024 Bit Static MOS RAM With Common I/O .....	7-19
5101	1024 Bit Static CMOS RAM .....	7-23
8212	Eight-Bit Input/Output Port .....	7-27
8255A	Programmable Peripheral Interface .....	7-37
8251	Programmable Communication Interface ...	7-59
8205	High Speed 1 Out of 8 Binary Decoder .....	7-73
8214	Priority Interrupt Control Unit .....	7-79
8216/8226	4-Bit Parallel Bi-Directional Bus Driver .....	7-83
8253	Programmable Interval Timer .....	7-89
8259	Programmable Interrupt Controller .....	7-101
8279	Programmable Peripheral Interface .....	7-117

# 8308

## 8192 BIT STATIC MOS READ ONLY MEMORY

### Organization — 1024 Words x 8 Bits

- Fast Access — 450 ns
- Directly Compatible with 8080 CPU at Maximum Processor Speed
- Two Chip Select Inputs for Easy Memory Expansion
- Directly TTL Compatible — All Inputs and Outputs
- Three State Output — OR-Tie Capability
- Fully Decoded
- Standard Power Supplies +12V DC, 5V DC

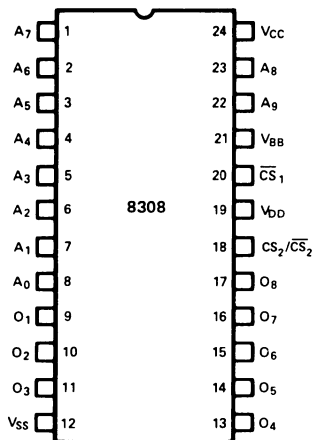
The Intel® 8308 is an 8,192 bit static MOS mask programmable Read Only Memory organized as 1024 words by 8-bits. This ROM is designed for 8080 microcomputer system applications where high performance, large bit storage, and simple interfacing are important design objectives. The inputs and outputs are fully TTL compatible.

A pin for pin compatible electrically programmed erasable ROM, the Intel® 8708, is available for system development and small quantity production use.

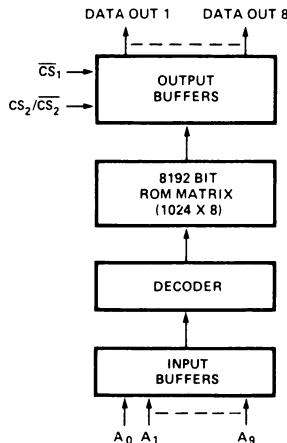
Two Chip Selects are provided —  $\overline{CS}_1$  which is negative true, and  $CS_2/\overline{CS}_2$  which may be programmed either negative or positive true at the mask level.

The 8308 read only memory is fabricated with N-channel silicon gate technology. This technology provides the designer with high performance, easy-to-use MOS circuits.

#### PIN CONFIGURATION



#### BLOCK DIAGRAM



#### PIN NAMES

$A_0$ - $A_9$	ADDRESS INPUTS
$O_1$ - $O_8$	DATA OUTPUTS
$\overline{CS}_1$ , $CS_2$	CHIP SELECT INPUTS

**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias . . . . .  $-25^{\circ}\text{C}$  to  $+85^{\circ}\text{C}$   
 Storage Temperature . . . . .  $-65^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$   
 Voltage On Any Pin With Respect  
     To  $V_{\text{BB}}$  . . . . .  $-0.3\text{V}$  to  $20\text{V}$   
 Power Dissipation . . . . . 1.0 Watt

**\*COMMENT**

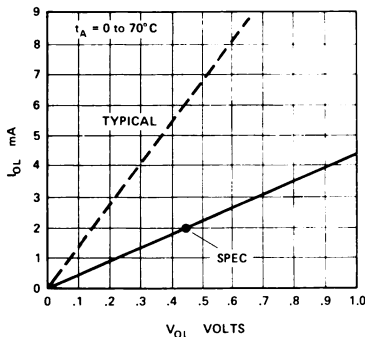
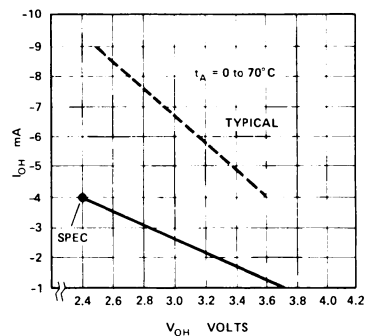
Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**D.C. AND OPERATING CHARACTERISTICS**

$T_A = 0^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$ ,  $V_{\text{CC}} = 5\text{V} \pm 5\%$ ;  $V_{\text{DD}} = 12\text{V} \pm 5\%$ ,  $V_{\text{BB}} = -5\text{V} \pm 5\%$ ,  $V_{\text{SS}} = 0\text{V}$  Unless Otherwise Specified.

Symbol	Parameter	Limits			Unit	Test Conditions
		Min.	Typ.[1]	Max.		
$I_{\text{LI}}$	Input Load Current (All Input Pins Except $\overline{\text{CS}}_1$ )			$\pm 10$	$\mu\text{A}$	$V_{\text{IN}} = 0$ to $5.25\text{V}$
$I_{\text{LCL}}$	Input Load Current on $\overline{\text{CS}}_1$			-1.6	mA	$V_{\text{IN}} = 0.45\text{V}$
$I_{\text{LPC}}$	Input Peak Load Current on $\overline{\text{CS}}_1$			-4	mA	$V_{\text{IN}} = 0.8\text{V}$ to $3.3\text{V}$
$I_{\text{LKC}}$	Input Leakage Current on $\overline{\text{CS}}_1$			10	$\mu\text{A}$	$V_{\text{IN}} = 3.3\text{V}$ to $5.25\text{V}$
$I_{\text{LO}}$	Output Leakage Current			10	$\mu\text{A}$	Chip Deselected
$V_{\text{IL}}$	Input "Low" Voltage	$V_{\text{SS}} - 1$		0.8V	V	
$V_{\text{IH}}$	Input "High" Voltage	3.3		$V_{\text{CC}} + 1.0$	V	
$V_{\text{OL}}$	Output "Low" Voltage			0.45	V	$I_{\text{OL}} = 2\text{mA}$
$V_{\text{OH1}}$	Output "High" Voltage	2.4			V	$I_{\text{OH}} = -4\text{mA}$
$V_{\text{OH2}}$	Output "High" Voltage	3.7			V	$I_{\text{OH}} = -1\text{mA}$
$I_{\text{CC}}$	Power Supply Current $V_{\text{CC}}$		.8	2	mA	
$I_{\text{DD}}$	Power Supply Current $V_{\text{DD}}$		32	60	mA	
$I_{\text{BB}}$	Power Supply Current $V_{\text{BB}}$		$10\mu\text{A}$	1	mA	
$P_{\text{D}}$	Power Dissipation			775	mW	

**NOTE 1:** Typical values for  $T_A = 25^{\circ}\text{C}$  and nominal supply voltage

**D.C. OUTPUT CHARACTERISTICS****D.C. OUTPUT CHARACTERISTICS**

## A.C. CHARACTERISTICS

$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$ ;  $V_{DD} = +12\text{V} \pm 5\%$ ,  $V_{BB} = -5\text{V} \pm 5\%$ ,  $V_{SS} = 0\text{V}$ , Unless Otherwise Specified.

Symbol	Parameter	Limits <sup>[2]</sup>			Unit
		Min.	Typ.	Max.	
$t_{ACC}$	Address to Output Delay Time		200	450	ns
$t_{CO1}$	Chip Select 1 to Output Delay Time		85	160	ns
$t_{CO2}$	Chip Select 2 to Output Delay Time		125	220	ns
$t_{DF}$	Chip Deselect to Output Data Float Time		125	220	ns

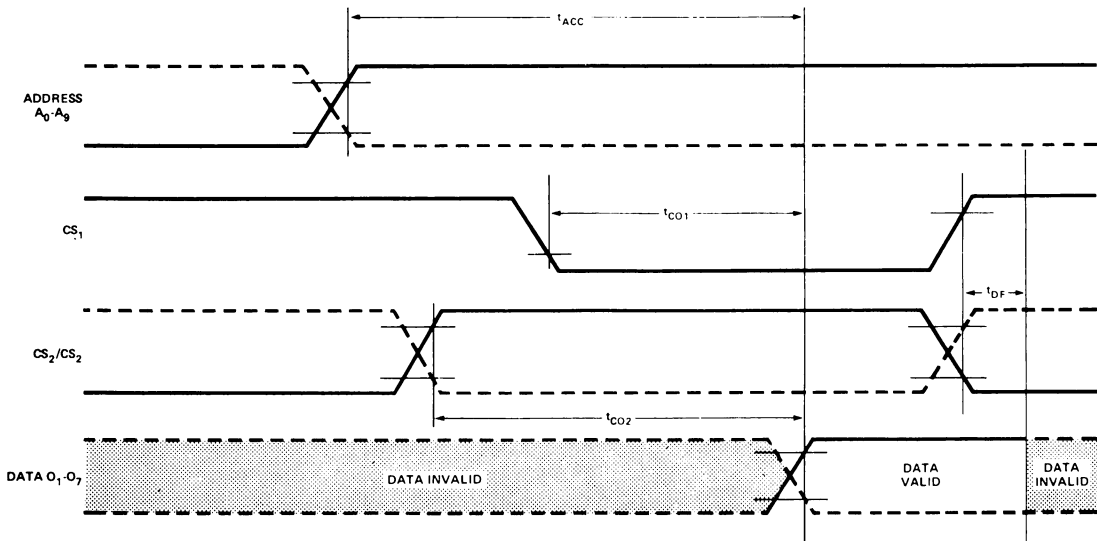
**NOTE 2:** Refer to conditions of Test for A.C. Characteristics. Add 50 nanoseconds (worst case) to specified values at  $V_{OH} = 3.7\text{V}$  @  $I_{OH} = -1\text{mA}$ ,  $C_L = 100\text{pF}$ .

## CONDITIONS OF TEST FOR A.C. CHARACTERISTICS

Output Load . . . . . 1 TTL Gate, and  $C_{LOAD} = 100\text{pF}$   
 Input Pulse Levels . . . . . .65V to 3.3V  
 Input Pulse Rise and Fall Times . . . . . 20 nsec  
 Timing Measurement Reference Level  
 . . . . . 2.4V  $V_{IH}$ ,  $V_{OH}$ ; 0.8V  $V_{IL}$ ,  $V_{OL}$

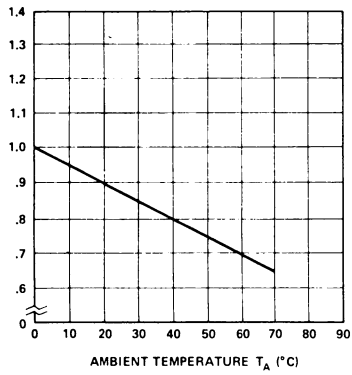
**CAPACITANCE**  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{MHz}$ ,  $V_{BB} = -5\text{V}$ ,  $V_{DD}$ ,  $V_{CC}$  and all other pins tied to  $V_{SS}$ .

Symbol	Test	Limits	
		Typ.	Max.
$C_{IN}$	Input Capacitance		6pF
$C_{OUT}$	Output Capacitance		12pF

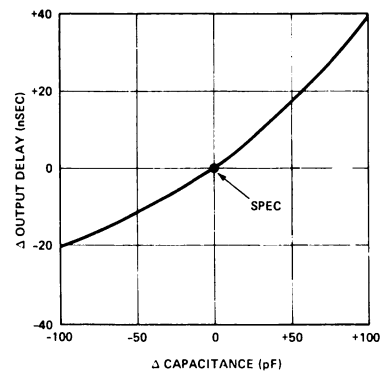


# **TYPICAL CHARACTERISTICS** (Nominal supply voltages unless otherwise noted.)

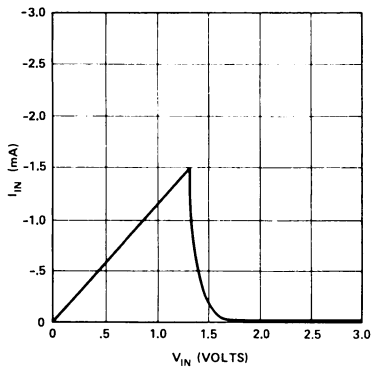
**$I_{DD}$  VS. TEMPERATURE  
(NORMALIZED)**



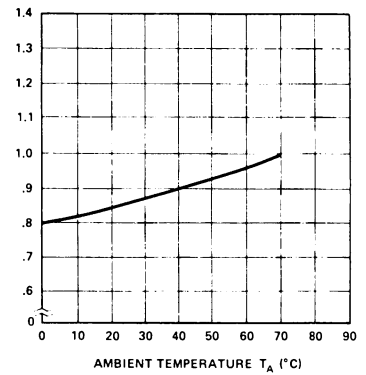
**$\Delta$  OUTPUT CAPACITANCE  
VS.  $\Delta$  OUTPUT DELAY**



**$\overline{CS}_1$  INPUT  
CHARACTERISTICS**



**$T_{ACC}$  VS. TEMPERATURE  
(NORMALIZED)**





## 8316A

# 16,384 BIT STATIC MOS READ ONLY MEMORY

Organization—2048 Words x 8 Bits

Access Time-850 ns max

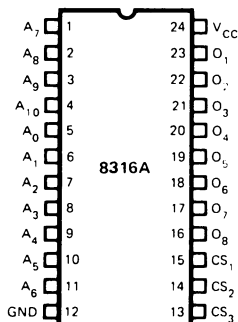
- Single +5 Volts Power Supply Voltage
- Directly TTL Compatible — All Inputs and Outputs
- Low Power Dissipation of 31.4  $\mu$ W/Bit Maximum
- Three Programmable Chip Select Inputs for Easy Memory Expansion
- Three-State Output — OR-Tie Capability
- Fully Decoded — On Chip Address Decode
- Inputs Protected — All Inputs Have Protection Against Static Charge

The Intel<sup>®</sup> 8316A is a 16,384-bit static MOS read only memory organized as 2048 words by 8 bits. This ROM is designed for microcomputer memory applications where high performance, large bit storage, and simple interfacing are important design objectives.

The inputs and outputs are fully TTL compatible. This device operates with a single +5V power supply. The three chip select inputs are programmable. Any combination of active high or low level chip select inputs can be defined and the desired chip select code is fixed during the masking process. These three programmable chip select inputs, as well as OR-tie compatibility on the outputs, facilitate easy memory expansion.

The 8316A read only memory is fabricated with N-channel silicon gate technology. This technology provides the designer with high performance, easy-to-use MOS circuits. Only a single +5V power supply is needed and all devices are directly TTL compatible.

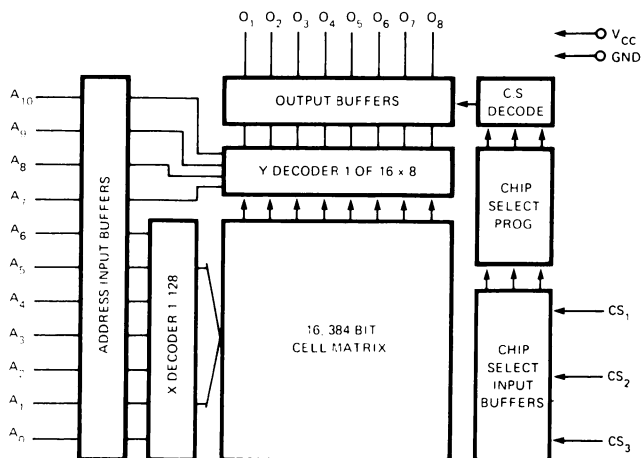
PIN CONFIGURATION



PIN NAMES

A <sub>0</sub> A <sub>10</sub>	ADDRESS INPUTS
O <sub>1</sub> O <sub>8</sub>	DATA OUTPUTS
CS <sub>1</sub> CS <sub>3</sub>	PROGRAMMABLE CHIP SELECT INPUTS

BLOCK DIAGRAM



**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias . . . . . 0°C to 70°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage On Any Pin With Respect  
   To Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1.0 Watt

\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**D.C. AND OPERATING CHARACTERISTICS**

$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$  unless otherwise specified

SYMBOL	PARAMETER	LIMITS			UNIT	TEST CONDITIONS
		MIN.	TYP. <sup>(1)</sup>	MAX.		
$I_{LI}$	Input Load Current (All Input Pins)			10	$\mu\text{A}$	$V_{IN} = 0$ to $5.25\text{V}$
$I_{LOH}$	Output Leakage Current			10	$\mu\text{A}$	$CS = 2.2\text{V}$ , $V_{OUT} = 4.0\text{V}$
$I_{LOL}$	Output Leakage Current			-20	$\mu\text{A}$	$CS = 2.2\text{V}$ , $V_{OUT} = 0.45\text{V}$
$I_{CC}$	Power Supply Current		40	98	mA	All inputs 5.25V Data Out Open
$V_{IL}$	Input "Low" Voltage	-0.5		0.8	V	
$V_{IH}$	Input "High" Voltage	2.0		$V_{CC} + 1.0\text{V}$	V	
$V_{OL}$	Output "Low" Voltage			0.45	V	$I_{OL} = 2.0\text{ mA}$
$V_{OH}$	Output "High" Voltage	2.2			V	$I_{OH} = -100\text{ }\mu\text{A}$

(1) Typical values for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.

**A.C. CHARACTERISTICS**

$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$  unless otherwise specified

SYMBOL	PARAMETER	LIMITS			UNIT
		MIN.	TYP. <sup>(1)</sup>	MAX.	
$t_A$	Address to Output Delay Time		400	850	nS
$t_{CO}$	Chip Select to Output Enable Delay Time			300	nS
$t_{DF}$	Chip Deselect to Output Data Float Delay Time	0		300	nS

**CONDITIONS OF TEST FOR  
A.C. CHARACTERISTICS**

Output Load . . . 1 TTL Gate, and  $C_{LOAD} = 100\text{ pF}$   
 Input Pulse Levels . . . . . 0.8 to 2.0V  
 Input Pulse Rise and Fall Times . (10% to 90%) 20 nS  
 Timing Measurement Reference Level  
   Input . . . . . 1.5V  
   Output . . . . . 0.45V to 2.2V

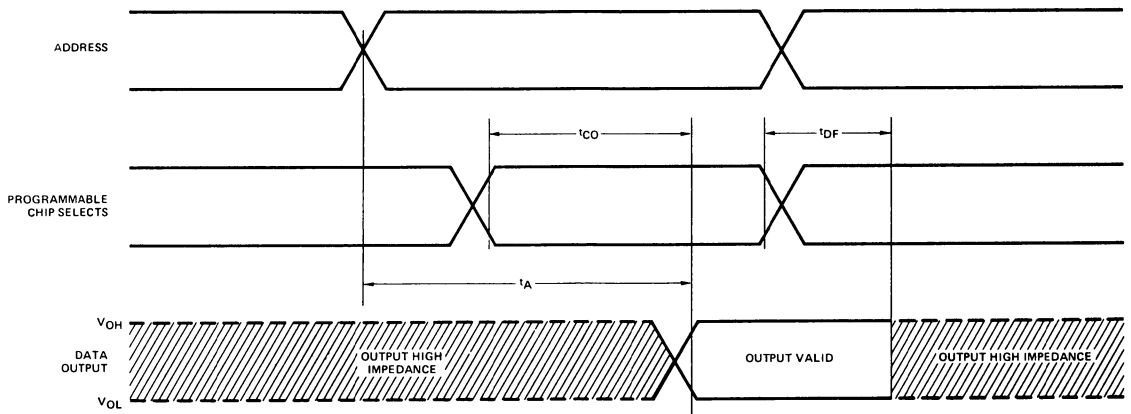
**CAPACITANCE<sup>(2)</sup>  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{ MHz}$** 

SYMBOL	TEST	LIMITS	
		TYP.	MAX.
$C_{IN}$	All Pins Except Pin Under Test Tied to AC Ground	4 pF	10 pF
$C_{OUT}$	All Pins Except Pin Under Test Tied to AC Ground	8 pF	15 pF

(2) This parameter is periodically sampled and is not 100% tested.



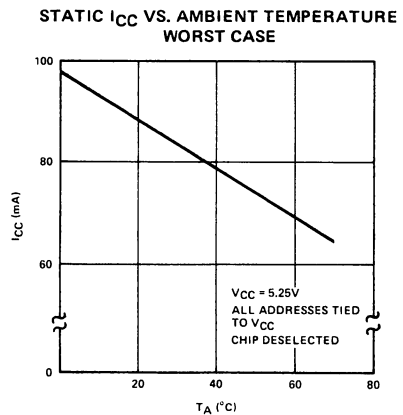
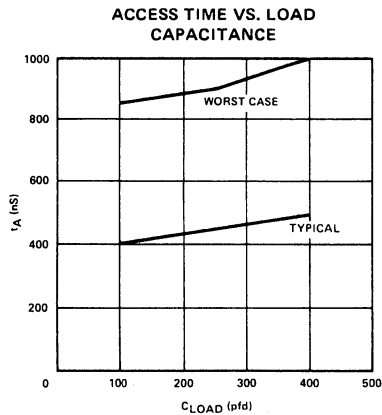
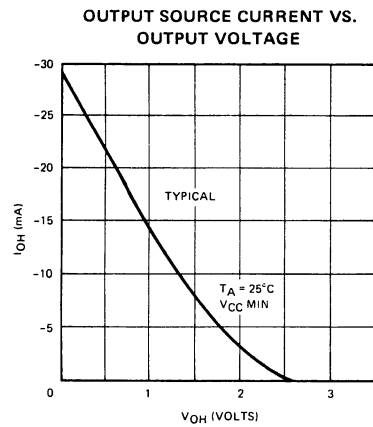
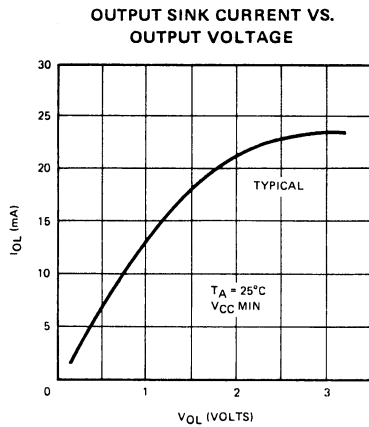
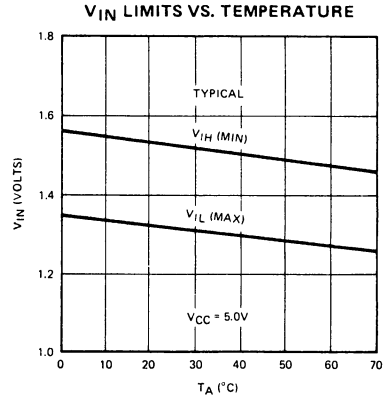
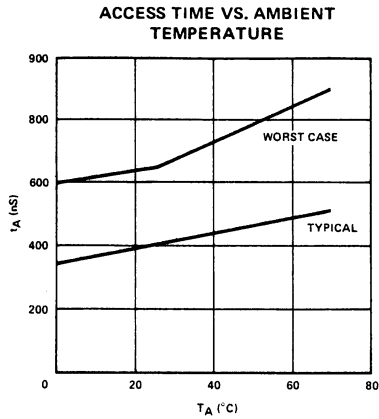
## WAVEFORMS



## 16K ROM PROTOTYPING

ROM systems may be developed and programs may be verified using Intel's 1702A or 2708 PROMs.

## TYPICAL D.C. CHARACTERISTICS





# MCS® CUSTOM ROM ORDER FORM

## 8316A ROM

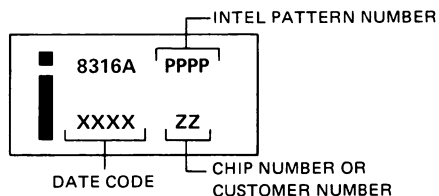
CUSTOMER _____	
P.O. NUMBER _____	
DATE _____	
For Intel use only	
S# _____	PPPP _____
STD _____	ZZ _____
_____	DD _____
APP _____	DATE _____

All custom 8316A ROM orders must be submitted on this form. Programming information should be sent in the form of computer punched cards or punched paper tape per the formats designated on this order form. Additional forms are available from Intel.

### MARKING

The marking as shown at the right must contain the Intel® logo, the product type (P8316A), the 4-digit Intel pattern number (PPPP), a date code (XXXX), and the 2-digit chip number (DD). An optional customer identification number may be substituted for the chip number (ZZ). Optional Customer Number (maximum 9 characters or spaces).

CUSTOMER NUMBER \_\_\_\_\_



### MASK OPTION SPECIFICATIONS

**A. CHIP NUMBER** \_\_\_\_\_ (Must be specified—any number from 0 through 7—DD).

The chip number will be coded in terms of positive logic where a logic "1" is a high level input.

Chip Number	CS3	CS2	CS1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

#### B. ROM Truth Table Format

Programming information should be sent in the form of computer punched cards or punched paper tape. In either case, a printout of the truth table should be accompanied with the order.

The following general format is applicable to the programming information sent to Intel:

- Data fields should be ordered beginning with the least significant address (0000) and ending with the most significant address (2047).
- A data field should start with the most significant bit and end with the least significant bit.

- The data field should consist of P's and N's. A P is to indicate a high level output (most positive) and an N a low level output (most negative). In terms of positive logic, a P is defined as a logic "1" and an N is defined as a logic "0". If the programming information is sent on a punched paper tape, then a start character, B, and an end character, F, must be used in the data field.

#### 1. Punched Card Format

An 80-column Hollerith card (preferably interpreted) punched by an IBM 026 or 029 keypunch should be submitted. The first card will be a title card; the format is as follows:

# MCS® CUSTOM ROM ORDER FORM

## a. Title Card

NO. OF OUTPUTS  
4 or 8

TITLE CARD DESIGNATION

CUSTOMER'S COMPANY NAME

CUSTOMER'S DIVISION OR LOCATION

CUSTOMER'S P/N

INTEL P/N

DECIMAL NUMBER INDICATING THE TRUTH TABLE NUMBER

Column	Data
1	Punch a T
2-5	Blank
6-30	Customer Company Name
31-34	Blank
35-54	Customer's Company Division or location
55-57	Blank
58-66	Customer Part Number
67	Blank
68-75	Punch the Intel 4-digit basic part number and in ( ) the number of output bits, e.g., 8316A(8).
76-78	Blank
79-80	Punch a 2-digit decimal number to identify the truth table number (mask programmed chip select number).

b. For a 2048 word X 8-bit organization only, cards 2 and the following cards should be punched as shown.

MSB (OUTPUT 8)

LSB (OUTPUT 1)

8 DATA FIELDS

DECIMAL WORD ADDRESS BEGINNING EACH CARD

DECIMAL NUMBER INDICATING THE TRUTH TABLE NUMBER

Column	Data
1-5	Punch the 5-digit decimal equivalent of the binary coded location which begins each card. The address is right justified, i.e., 00000, 00008, 00016, etc.
6	Blank
7-14	Data Field
15	Blank
16-23	Data Field
33	Blank
34-41	Data Field
42	Blank
43-50	Data Field
51	Blank
52-59	Data Field
60	Blank
61-68	Data Field
69	Blank
70-77	Data Field
78	Blank
79-80	Punch same 2-digit decimal number as in title card.

## 2. Paper Tape Format

1" wide paper tape using 7- or 8-bit ASCII code, such as a model 33 ASR Teletype produces, or the 11/16" wide paper tape using a 5-bit Baudot code, such as a Telex produces.

The format requirements are as follows:

- All word fields are to be punched in consecutive order, starting with word field 0 (all addresses low). There must be exactly 2048 word fields for the 2048 X 8 ROM organization.
- Each word field must begin with the start character B and end with the stop character F. There must be exactly 8 data characters between the B and F.

NO OTHER CHARACTERS, SUCH AS RUBOUTS, ARE ALLOWED ANYWHERE IN A WORD FIELD. If in preparing a tape an error is made, the entire word field, including the B and F, must be rubbed out. Within the word field, a P results in a high level output and an N results in a low level output.

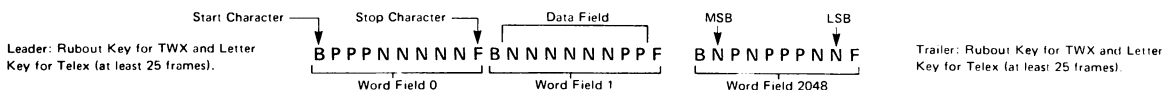
- Preceding the first word field and following the last word field, there must be a leader/trailer length of at least 25 characters. This should consist of rubout or null punches (letter key for Telex tapes).

- Between word fields, comments not containing B's or F's may be inserted. Carriage return and line feed characters should be inserted as a "comment"

just before each word field (or at least between every four word fields). When these carriage returns, etc., are inserted, the tape may be easily listed on the teletype for purposes of error checking. The customer may also find it helpful to insert the word number (as a comment) at least every four word fields.

- Included in the tape before the leader should be the customer's complete Telex or TWX number and, if more than one pattern is being transmitted, the ROM pattern number.

- MSB and LSB are the most and least significant bit of the device outputs. Refer to the data sheet for the pin numbers.





## 8708

# 8192 BIT ERASABLE AND ELECTRICALLY REPROGRAMMABLE READ ONLY MEMORY

### • 8708 1024x8 Organization

- **Fast Programming** —  
Typ. 100 sec. For All 8K Bits
- **Low Power During Programming**
- **Access Time** — 450 ns
- **Standard Power Supplies** —  
+12V,  $\pm 5V$
- **Static** — No Clocks Required
- **Inputs and Outputs TTL Compatible** During Both Read and Program Modes
- **Three-State Output** — OR-Tie Capability

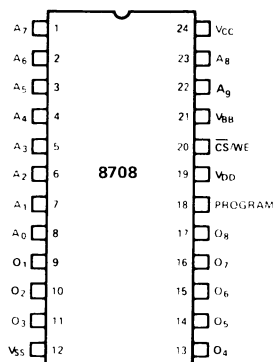
The Intel® 8708 is a high speed 8192 bit erasable and electrically reprogrammable ROM (EPROM) ideally suited where fast turn around and pattern experimentation are important requirements.

The 8708 is packaged in a 24 pin dual-in-line package with transparent lid. The transparent lid allows the user to expose the chip to ultraviolet light to erase the bit pattern. A new pattern can then be written into the device.

A pin for pin mask programmed ROM, the Intel® 8308, is available for large volume production runs of systems initially using the 8708.

The 8708 is fabricated with the time proven N-channel silicon gate technology.

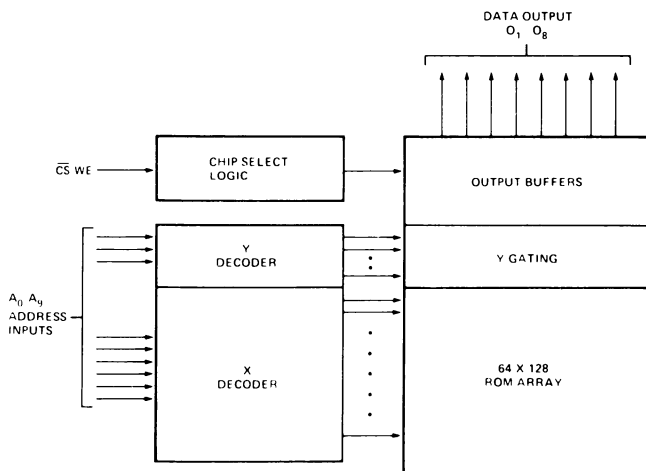
#### PIN CONFIGURATION



#### PIN NAMES

A <sub>0</sub> -A <sub>9</sub>	ADDRESS INPUTS
O <sub>1</sub> -O <sub>8</sub>	DATA OUTPUTS
CS/WE	CHIP SELECT/WRITE ENABLE INPUT

#### BLOCK DIAGRAM



## ABSOLUTE MAXIMUM RATINGS\*

Temperature Under Bias	-25°C to +85°C
Storage Temperature	-65°C to +125°C
All Input or Output Voltages with Respect to $V_{BB}$ (except Program)	+15V to -0.3V
Program Input to $V_{BB}$	+35V to -0.3V
Supply Voltages $V_{CC}$ and $V_{SS}$ with Respect to $V_{BB}$	+15V to -0.3V
$V_{DD}$ with Respect to $V_{BB}$	+20V to -0.3V
Power Dissipation	1.5W

### \*COMMENT

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## READ OPERATION

### D.C. AND OPERATING CHARACTERISTICS

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$ ,  $V_{DD} = +12\text{V} \pm 5\%$ ,  $V_{BB} = -5\text{V} \pm 5\%$ ,  $V_{SS} = 0\text{V}$ , Unless Otherwise Noted.

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Conditions
$I_{LI}$	Address and Chip Select Input Load Current			10	$\mu\text{A}$	$V_{IN} = 5.25\text{V}$
$I_{LO}$	Output Leakage Current			10	$\mu\text{A}$	$V_{OUT} = 5.25\text{V}$ , $\overline{CS}/\overline{WE} = 5\text{V}$
$I_{DD}$	$V_{DD}$ Supply Current		50	65	mA	Worst Case Supply Currents: All Inputs High $\overline{CS}/\overline{WE} = 5\text{V}$ ; $T_A = 0^\circ\text{C}$
$I_{CC}$	$V_{CC}$ Supply Current		6	10	mA	
$I_{BB}$	$V_{BB}$ Supply Current		30	45	mA	
$V_{IL}$	Input Low Voltage	$V_{SS}$		0.65	V	
$V_{IH}$	Input High Voltage	3.0		$V_{CC}+1$	V	
$V_{OL}$	Output Low Voltage			0.45	V	$I_{OL} = 1.6\text{mA}$
$V_{OH1}$	Output High Voltage	3.7			V	$I_{OH} = -100\mu\text{A}$
$V_{OH2}$	Output High Voltage	2.4			V	$I_{OH} = -1\text{mA}$
$P_D$	Power Dissipation			800	mW	$T_A = 70^\circ\text{C}$

NOTES: 1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltages.

2. The program input (Pin 18) may be tied to  $V_{SS}$  or  $V_{CC}$  during the read mode.

## A.C. CHARACTERISTICS

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$ ,  $V_{DD} = +12\text{V} \pm 5\%$ ,  $V_{BB} = -5\text{V} \pm 5\%$ ,  $V_{SS} = 0\text{V}$ , Unless Otherwise Noted.

Symbol	Parameter	Min.	Typ.	Max.	Unit
$t_{ACC}$	Address to Output Delay		280	450	ns
$t_{CO}$	Chip Select to Output Delay			120	ns
$t_{DF}$	Chip De-Select to Output Float	0		120	ns
$t_{OH}$	Address to Output Hold	0			ns

Capacitance<sup>[1]</sup>  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{MHz}$

Symbol	Parameter	Typ.	Max.	Unit	Conditions
$C_{IN}$	Input Capacitance	4	6	pF	$V_{IN}=0\text{V}$
$C_{OUT}$	Output Capacitance	8	12	pF	$V_{OUT}=0\text{V}$

Note 1. This parameter is periodically sampled and not 100% tested.

### A.C. Test Conditions:

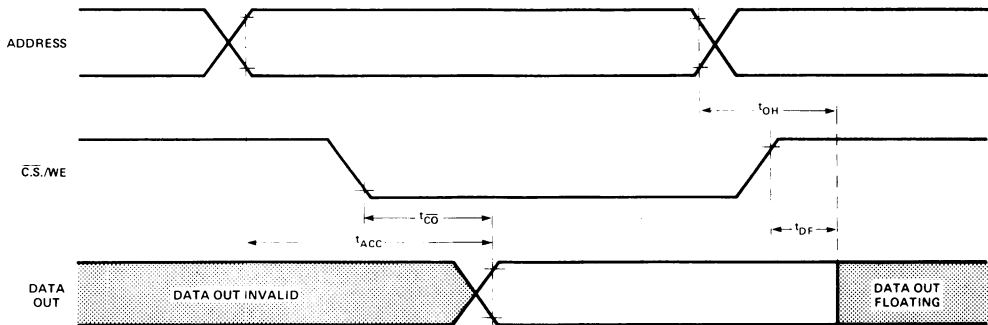
Output Load: 1 TTL gate and  $C_L = 100\text{pF}$

Input Rise and Fall Times:  $\leq 20\text{ns}$

Timing Measurement Reference Levels: 0.8V and 2.8V for inputs; 0.8V and 2.4V for outputs

Input Pulse Levels: 0.65V to 3.0V

### Waveforms



# STANDARD FORM NO. 7-14

STANDARD FORM NO. 7-14

DATE	TIME	LOCATION	REMARKS
10/1/54	10:00	1000	1000
10/1/54	10:00	1000	1000
10/1/54	10:00	1000	1000
10/1/54	10:00	1000	1000
10/1/54	10:00	1000	1000

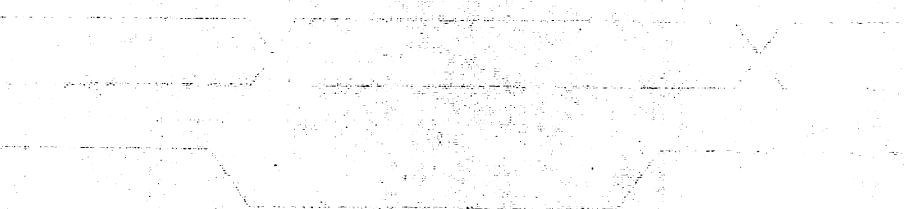
## STANDARD FORM NO. 7-14

STANDARD FORM NO. 7-14

## STANDARD FORM NO. 7-14

STANDARD FORM NO. 7-14

## STANDARD FORM NO. 7-14





# 8101A-4

## 1024 BIT STATIC MOS RAM

### WITH SEPARATE I/O

\* 450 nsec Access Time Maximum

\* 256 Word by 4 Bit Organization

- Single +5V Supply Voltage
- Directly TTL Compatible: All Inputs and Outputs
- Static MOS: No Clocks or Refreshing Required
- Simple Memory Expansion: Chip Enable Input
- Powerful Output Drive Capability
- Low Cost Packaging: 22 Pin Plastic Dual In-Line Configuration
- Low Power: Typically 150mW
- Three-State Output: OR-Tie Capability
- Output Disable Provided for Ease of Use in Common Data Bus Systems

The Intel® 8101A-4 is a 256 word by 4-bit static random access memory element using N-channel MOS devices integrated on a monolithic array. It uses fully DC stable (static) circuitry and therefore requires no clocks or refreshing to operate. The data is read out nondestructively and has the same polarity as the input data.

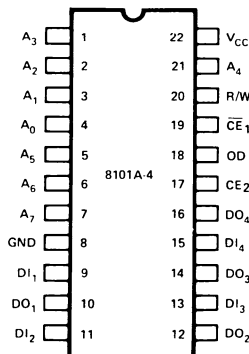
The 8101A-4 is designed for memory applications where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. Two chip-enables allow easy selection of an individual package when outputs are OR-tied. An output disable is provided so that data inputs and outputs can be tied for common I/O systems. The output disable function eliminates the need for bi-directional logic in a common I/O system.

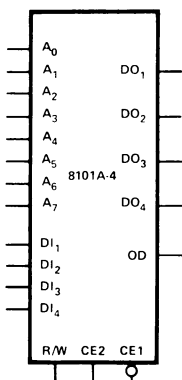
The Intel® 8101A-4 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

Intel's silicon gate technology also provides excellent protection against contamination. This permits the use of low cost plastic packaging.

#### PIN CONFIGURATION



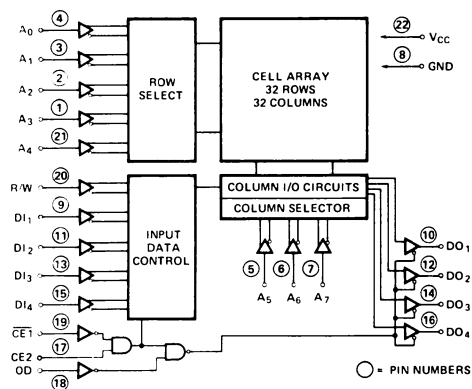
#### LOGIC SYMBOL



#### PIN NAMES

DI <sub>1</sub> , DI <sub>4</sub>	DATA INPUT	CE <sub>2</sub>	CHIP ENABLE 2
A <sub>0</sub> , A <sub>7</sub>	ADDRESS INPUTS	OD	OUTPUT DISABLE
R/W	READ/WRITE INPUT	DO <sub>1</sub> , DO <sub>4</sub>	DATA OUTPUT
CE <sub>1</sub>	CHIP ENABLE 1	V <sub>CC</sub>	POWER (+5V)

#### BLOCK DIAGRAM



**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias . . . . . -10°C to 80°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage On Any Pin  
     With Respect to Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

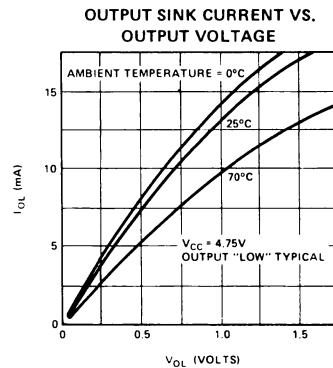
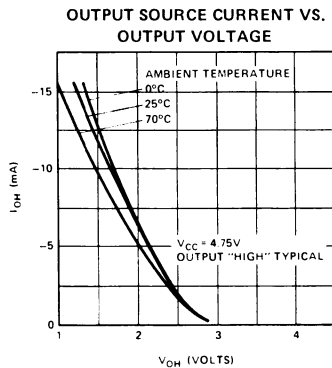
**\*COMMENT:**

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

**D.C. AND OPERATING CHARACTERISTICS**

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$  unless otherwise specified.

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$I_{LI}$	Input Current		1	10	$\mu\text{A}$	$V_{IN} = 0$ to $5.25\text{V}$
$I_{LOH}$	I/O Leakage Current <sup>[2]</sup>		1	10	$\mu\text{A}$	Output Disabled, $V_{OUT} = 4.0\text{V}$
$I_{LOL}$	I/O Leakage Current <sup>[2]</sup>		-1	-10	$\mu\text{A}$	Output Disabled, $V_{OUT} = 0.45\text{V}$
$I_{CC1}$	Power Supply Current		35	55	mA	$V_{IN} = 5.25\text{V}$ , $I_O = 0\text{mA}$ $T_A = 25^\circ\text{C}$
$I_{CC2}$	Power Supply Current			60	mA	$V_{IN} = 5.25\text{V}$ , $I_O = 0\text{mA}$ $T_A = 0^\circ\text{C}$
$V_{IL}$	Input "Low" Voltage	-0.5		+0.8	V	
$V_{IH}$	Input "High" Voltage	2.0		$V_{CC}$	V	
$V_{OL}$	Output "Low" Voltage			+0.45	V	$I_{OL} = 2.0\text{mA}$
$V_{OH}$	Output "High" Voltage	2.4			V	$I_{OH} = -400\mu\text{A}$

**TYPICAL D.C. CHARACTERISTICS**

NOTES: 1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.  
 2. Input and Output tied together.

## A.C. CHARACTERISTICS

**READ CYCLE**  $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5V \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$t_{RC}$	Read Cycle	450			ns	(See Below)
$t_A$	Access Time			450	ns	
$t_{CO}$	Chip Enable To Output			310	ns	
$t_{OD}$	Output Disable To Output			250	ns	
$t_{DF}$ [2]	Data Output to High Z State	0		200	ns	
$t_{OH}$	Previous Read Data Valid after change of Address	40			ns	

## WRITE CYCLE

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$t_{WC}$	Write Cycle	270			ns	(See Below)
$t_{AW}$	Write Delay	20			ns	
$t_{CW}$	Chip Enable To Write	250			ns	
$t_{DW}$	Data Setup	250			ns	
$t_{DH}$	Data Hold	0			ns	
$t_{WP}$	Write Pulse	250			ns	
$t_{WR}$	Write Recovery	0			ns	
$t_{DS}$	Output Disable Setup	20			ns	

## A.C. CONDITIONS OF TEST

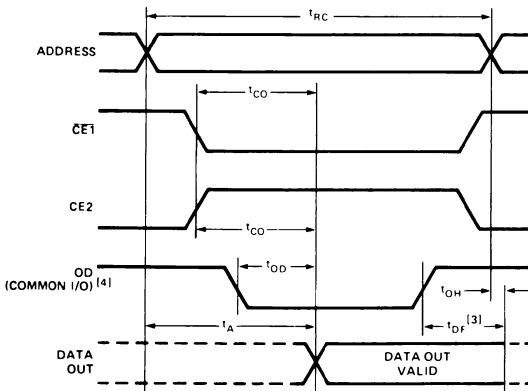
$t_r, t_f$  ..... 20 ns  
 Input Levels ..... 0.8V or 2.0V  
 Timing Reference ..... 1.5V  
 Load ..... 1 TTL Gate and  $C_L = 100$  pF

## CAPACITANCE<sup>[3]</sup> $T_A = 25^\circ\text{C}$ , $f = 1$ MHz

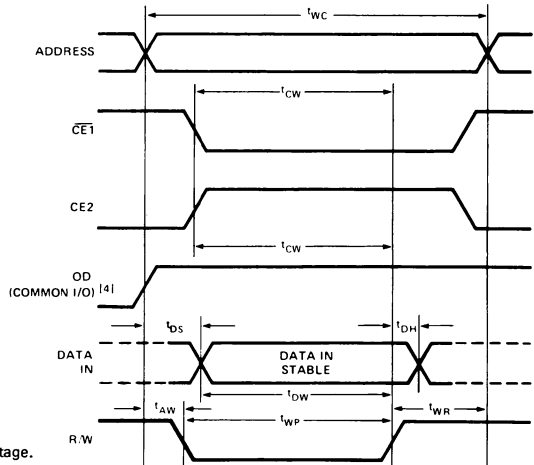
Symbol	Test	Limits (pF)	
		Typ. <sup>[1]</sup>	Max.
$C_{IN}$	Input Capacitance (All Input Pins) $V_{IN} = 0V$	4	8
$C_{OUT}$	Output Capacitance $V_{OUT} = 0V$	8	12

## WAVEFORMS

### READ CYCLE



### WRITE CYCLE



- NOTES:
1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.
  2.  $t_{DF}$  is with respect to the trailing edge of  $\overline{CE}_1$ ,  $\overline{CE}_2$ , or  $\overline{OD}$ , whichever occurs first.
  3. This parameter is periodically sampled and is not 100% tested.

4.  $\overline{OD}$  should be tied low for separate I/O operation.



# 8111A-4

## 1024 BIT STATIC MOS RAM WITH COMMON I/O

- \* 450 nsec Access Time Maximum
- \* 256 Word by 4 Bit Organization

- Single +5V Supply Voltage
- Directly TTL Compatible: All Inputs and Outputs
- Static MOS: No Clocks or Refreshing Required
- Simple Memory Expansion: Chip Enable Input
- Powerful Output Drive Capability
- Low Cost Packaging: 18 Pin Plastic Dual In-Line Configuration
- Low Power: Typically 150mW
- Three-State Output: OR-Tie Capability
- Output Disable Provided for Ease of Use in Common Data Bus Systems

The Intel® 8111A-4 is a 256 word by 4-bit static random access memory element using N-channel MOS devices integrated on a monolithic array. It uses fully DC stable (static) circuitry and therefore requires no clocks or refreshing to operate. The data is read out nondestructively and has the same polarity as the input data. Common input/output pins are provided.

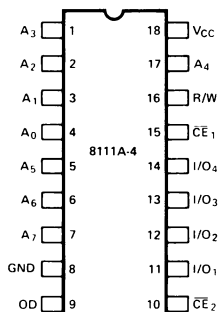
The 8111A-4 is designed for memory applications in small systems where high performance, low cost, large bit storage, and simple interfacing are important design objectives.

It is directly TTL compatible in all respects: inputs, outputs, and a single +5V supply. Separate chip enable ( $\overline{CE}$ ) leads allow easy selection of an individual package when outputs are OR-tied.

The Intel® 8111A-4 is fabricated with N-channel silicon gate technology. This technology allows the design and production of high performance, easy-to-use MOS circuits and provides a higher functional density on a monolithic chip than either conventional MOS technology or P-channel silicon gate technology.

Intel's silicon gate technology also provides excellent protection against contamination. This permits the use of low cost plastic packaging.

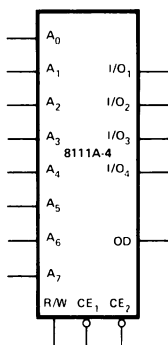
### PIN CONFIGURATION



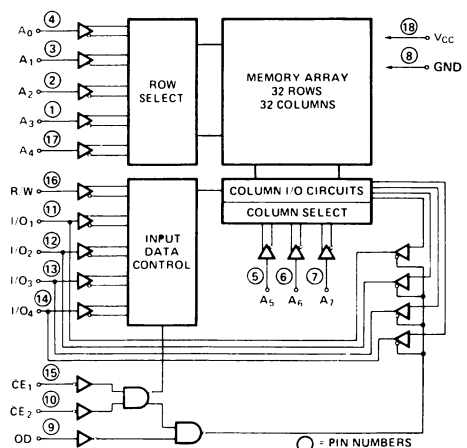
### PIN NAMES

A <sub>0</sub> -A <sub>7</sub>	ADDRESS INPUTS
OD	OUTPUT DISABLE
R/W	READ/WRITE INPUT
CE <sub>1</sub>	CHIP ENABLE 1
CE <sub>2</sub>	CHIP ENABLE 2
I/O <sub>1</sub> -I/O <sub>4</sub>	DATA INPUT/OUTPUT

### LOGIC SYMBOL



### BLOCK DIAGRAM



**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias . . . . . -10°C to 80°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage On Any Pin  
     With Respect to Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

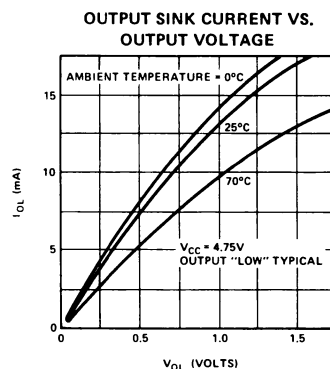
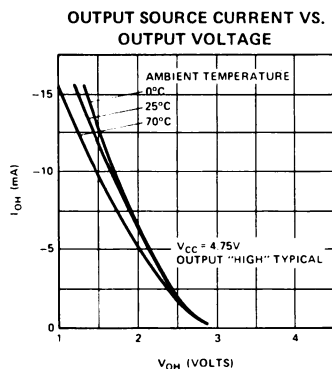
**\*COMMENT:**

*Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

**D.C. AND OPERATING CHARACTERISTICS**

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$I_{LI}$	Input Load Current		1	10	$\mu\text{A}$	$V_{IN} = 0$ to $5.25\text{V}$
$I_{LOH}$	I/O Leakage Current		1	10	$\mu\text{A}$	Output Disabled, $V_{I/O} = 4.0\text{V}$
$I_{LOL}$	I/O Leakage Current		-1	-10	$\mu\text{A}$	Output Disabled, $V_{I/O} = 0.45\text{V}$
$I_{CC1}$	Power Supply Current		35	55	mA	$V_{IN} = 5.25\text{V}$ $I_{I/O} = 0\text{mA}$ , $T_A = 25^\circ\text{C}$
$I_{CC2}$	Power Supply Current			60	mA	$V_{IN} = 5.25\text{V}$ $I_{I/O} = 0\text{mA}$ , $T_A = 0^\circ\text{C}$
$V_{IL}$	Input Low Voltage	-0.5		0.8	V	
$V_{IH}$	Input High Voltage	2.0		$V_{CC}$	V	
$V_{OL}$	Output Low Voltage			0.45	V	$I_{OL} = 2.0\text{mA}$
$V_{OH}$	Output High Voltage	2.4			V	$I_{OH} = -400\mu\text{A}$



NOTE: 1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.

## A.C. CHARACTERISTICS

READ CYCLE  $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$t_{RC}$	Read Cycle	450			ns	(See Below)
$t_A$	Access Time			450	ns	
$t_{CO}$	Chip Enable To Output			310	ns	
$t_{OD}$	Output Disable To Output			250	ns	
$t_{DF}$ [2]	Data Output to High Z State	0		200	ns	
$t_{OH}$	Previous Read Data Valid after change of Address	40			ns	

## WRITE CYCLE

Symbol	Parameter	Min.	Typ. <sup>[1]</sup>	Max.	Unit	Test Conditions
$t_{WC}$	Write Cycle	270			ns	(See Below)
$t_{AW}$	Write Delay	20			ns	
$t_{CW}$	Chip Enable To Write	250			ns	
$t_{DW}$	Data Setup	250			ns	
$t_{DH}$	Data Hold	0			ns	
$t_{WP}$	Write Pulse	250			ns	
$t_{WR}$	Write Recovery	0			ns	
$t_{DS}$	Output Disable Setup	20			ns	

## A.C. CONDITIONS OF TEST

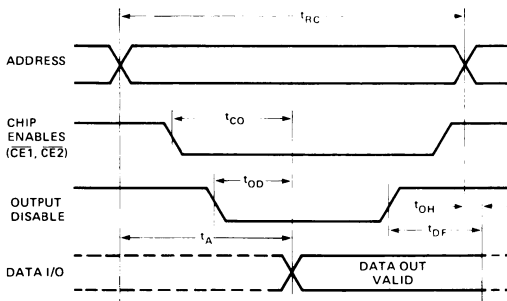
$t_r, t_f$  ..... 20 ns  
 Input Levels ..... 0.8V or 2.0V  
 Timing Reference ..... 1.5V  
 Load ..... 1 TTL Gate and  $C_L = 100$  pF

## CAPACITANCE<sup>[3]</sup> $T_A = 25^\circ\text{C}$ , $f = 1\text{MHz}$

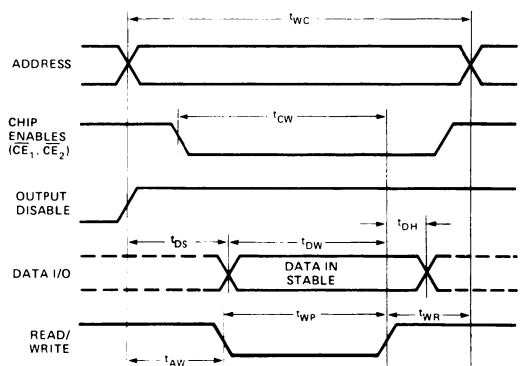
Symbol	Test	Limits (pF)	
		Typ. <sup>[1]</sup>	Max.
$C_{IN}$	Input Capacitance (All Input Pins) $V_{IN} = 0\text{V}$	4	8
$C_{I/O}$	I/O Capacitance $V_{I/O} = 0\text{V}$	10	15

## WAVEFORMS

### READ CYCLE



### WRITE CYCLE



- NOTES:
1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.
  2.  $t_{DF}$  is with respect to the trailing edge of  $\overline{CE}_1$ ,  $\overline{CE}_2$ , or  $\overline{OD}$ , whichever occurs first.
  3. This parameter is periodically sampled and is not 100% tested.







# 5101, 5101L

## 1024 BIT (256 x 4) STATIC CMOS RAM

P/N	Typ. Current @ 2V ( $\mu$ A)	Typ. Standby Current ( $\mu$ A)	Max Access (ns)
5101L	0.14	0.2	650
5101L-1	0.9	1.5	450
5101L-3	0.7	1.0	650
5101-1	—	1.5	450
5101	—	0.2	650
5101-3	—	1.0	650
5101-8	—	10.0	800

- Single +5V Power Supply
- Ideal for Battery Operation (5101L)
- Directly TTL Compatible: All Inputs and Outputs
- Three-State Output

The Intel® 5101 and 5101L are ultra-low power 1024 bit (256 words x 4-bits) static RAMs fabricated with an advanced ion-implanted silicon gate CMOS technology. The devices have two chip enable inputs. Minimum standby current is drawn by these devices when  $CE_2$  is at a low level. When deselected the 5101 draws from the single 5 volt supply only 15 microamps. These devices are ideally suited for low power applications where battery operation or battery backup for non-volatility are required.

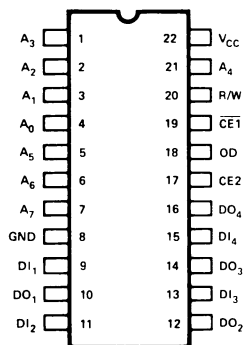
The 5101 uses fully DC stable (static) circuitry; it is not necessary to pulse chip select for each address transition. The data is read out non-destructively and has the same polarity as the input data. All inputs and outputs are directly TTL compatible. The 5101 has separate data input and data output terminals. An output disable function is provided so that the data inputs and outputs may be wire OR-ed for use in common data I/O systems.

*The 5101L is identical to the 5101 with the additional feature of guaranteed data retention at a power supply voltage as low as 2.0 volts.*

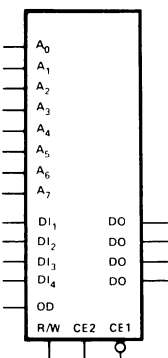
A pin compatible N-channel static RAM, the Intel® 2101A, is also available for low cost applications where a 256 x 4 organization is needed.

The Intel ion-implanted, silicon gate, complementary MOS (CMOS) allows the design and production of ultra-low power, high performance memories.

### PIN CONFIGURATION



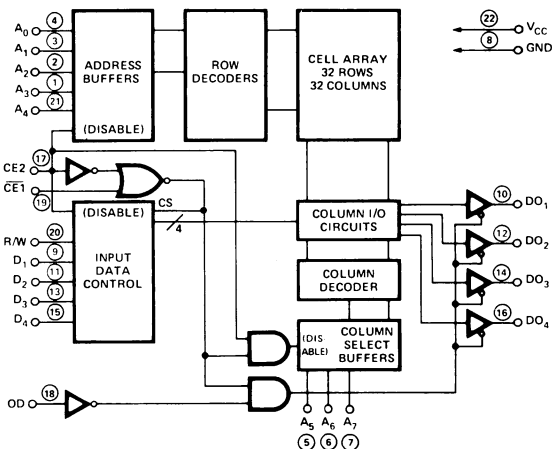
### LOGIC SYMBOL



### TRUTH TABLE

$CE_1$	$CE_2$	OD	R/W	$D_{IN}$	Output	Mode
H	X	X	X	X	High Z	Not Selected
X	L	X	X	X	High Z	Not Selected
X	X	H	H	X	High Z	Output Disabled
L	H	H	L	X	High Z	Write
L	H	L	X	$D_{IN}$	Write	Write
L	H	L	H	X	$D_{OUT}$	Read

### BLOCK DIAGRAM



○ = PIN NUMBERS

## Absolute Maximum Ratings \*

Ambient Temperature Under Bias . . . . . -10°C to 80°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage On Any Pin  
     With Respect to Ground . . . . -0.3V to V<sub>CC</sub> +0.3V  
 Maximum Power Supply Voltage . . . . . +7.0V  
 Power Dissipation . . . . . 1 Watt

\*COMMENT:

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## D. C. and Operating Characteristics

T<sub>A</sub> = 0°C to 70°C, V<sub>CC</sub> = 5V ±5% unless otherwise specified.

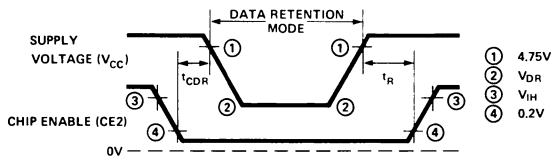
Symbol	Parameter	5101 (Except 5101-8) and 5101L Family Limits			5101-8 Limits			Units	Test Conditions
		Min.	Typ. <sup>[1]</sup>	Max.	Min.	Typ. <sup>[1]</sup>	Max.		
I <sub>LI</sub> <sup>[2]</sup>	Input Current		5			5		nA	
I <sub>LO</sub>   <sup>[2]</sup>	Output Leakage Current			1			2	μA	CE1 = 2.2V, V <sub>OUT</sub> = 0 to V <sub>CC</sub>
I <sub>CC1</sub>	Operating Current		9	22		11	25	mA	V <sub>IN</sub> = V <sub>CC</sub> , Except CE1 ≤ 0.65V, Outputs Open
I <sub>CC2</sub>	Operating Current		13	27		15	30	mA	V <sub>IN</sub> = 2.2V, Except CE1 ≤ 0.65V, Outputs Open
I <sub>CCL1</sub> <sup>[2]</sup>	5101 and 5101-1 Standby Current			15			—	μA	CE2 ≤ 0.2V, V <sub>CC</sub> = 5V ±5%
I <sub>CCL2</sub> <sup>[2]</sup>	5101-3 Standby Current		1	200			—	μA	CE2 ≤ 0.2V, V <sub>CC</sub> = 5V ±5%
I <sub>CCL3</sub> <sup>[2]</sup>	5101-8 Standby Current			—		10	50	μA	CE2 ≤ 0.2V, V <sub>CC</sub> = 5V ±5%, T <sub>A</sub> = 25°C
I <sub>CCL4</sub> <sup>[2]</sup>	5101-8 Standby Current			—			500	μA	CE2 ≤ 0.2V, V <sub>CC</sub> = 5V ±5%, T <sub>A</sub> = 70°C
V <sub>IL</sub>	Input Low Voltage	-0.3		0.65	-0.3		0.65	V	
V <sub>IH</sub>	Input High Voltage	2.2		V <sub>CC</sub>	2.2		V <sub>CC</sub>	V	
V <sub>OL</sub>	Output Low Voltage			0.4			0.4	V	I <sub>OL</sub> = 2.0mA
V <sub>OH</sub>	Output High Voltage	2.4			2.4			V	I <sub>OH</sub> = 1.0mA

### Low V<sub>CC</sub> Data Retention Characteristics (For 5101L, 5101L-1, and 5101L-3) T<sub>A</sub> = 0°C to 70°C

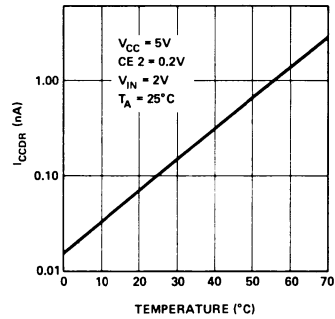
Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions	
V <sub>DR</sub>	V <sub>CC</sub> for Data Retention	2.0			V	CE2 ≤ 0.2V	
I <sub>CCDR1</sub>	5101L or 5101L-1 Data Retention Current		0.14	15	μA		V <sub>DR</sub> = 2.0V
I <sub>CCDR2</sub>	5101L-3 Data Retention Current		0.7	200	μA		V <sub>DR</sub> = 2.0V
t <sub>CDR</sub>	Chip Deselect to Data Retention Time	0			ns		
t <sub>R</sub>	Operation Recovery Time	t <sub>RC</sub> <sup>[3]</sup>			ns		

NOTES: 1. Typical values are T<sub>A</sub> = 25°C and nominal supply voltage. 2. Current through all inputs and outputs included in I<sub>CCL</sub> measurement. 3. t<sub>RC</sub> = Read Cycle Time.

## Low $V_{CC}$ Data Retention Waveform



## Typical $I_{CCDR}$ Vs. Temperature



## A.C. Characteristics $T_A = 0^\circ C$ to $70^\circ C$ , $V_{CC} = 5V \pm 5\%$ , unless otherwise specified.

### READ CYCLE

Symbol	Parameter	5101-1, 5101L-1 Limits (ns)		5101, 5101-3, 5101L and 5101L-3 Limits (ns)		5101-8 Limits (ns)	
		Min.	Max.	Min.	Max.	Min.	Max.
$t_{RC}$	Read Cycle	450		650		800	
$t_A$	Access Time		450		650		800
$t_{CO1}$	Chip Enable ( $\overline{CE\ 1}$ ) to Output		400		600		800
$t_{CO2}$	Chip Enable ( $\overline{CE\ 2}$ ) to Output		500		700		850
$t_{OD}$	Output Disable to Output		250		350		450
$t_{DF}$	Data Output to High Z State	0	130	0	150	0	200
$t_{OH1}$	Previous Read Data Valid with Respect to Address Change	0		0		0	
$t_{OH2}$	Previous Read Data Valid with Respect to Chip Enable	0		0		0	

### WRITE CYCLE

$t_{WC}$	Write Cycle	450	650	800
$t_{AW}$	Write Delay	130	150	200
$t_{CW1}$	Chip Enable ( $\overline{CE\ 1}$ ) to Write	350	550	650
$t_{CW2}$	Chip Enable ( $\overline{CE\ 2}$ ) to Write	350	550	650
$t_{DW}$	Data Setup	250	400	450
$t_{DH}$	Data Hold	50	100	100
$t_{WP}$	Write Pulse	250	400	450
$t_{WR}$	Write Recovery	50	50	100
$t_{DS}$	Output Disable Setup	130	150	200

## A. C. CONDITIONS OF TEST

Input Pulse Levels: +0.65 Volt to 2.2 Volt

Input Pulse Rise and Fall Times: 20nsec

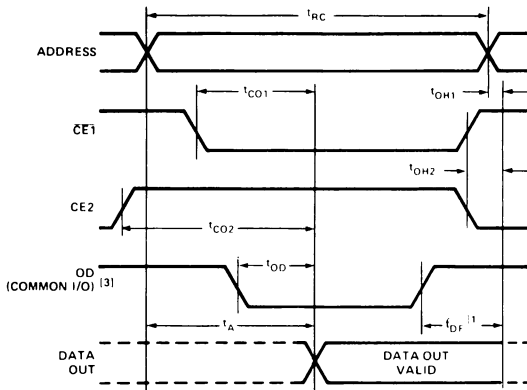
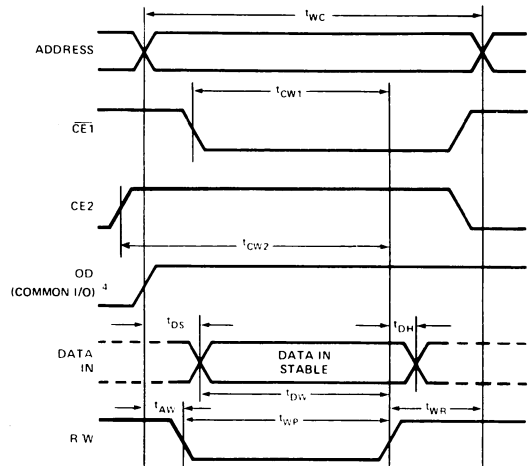
Timing Measurement Reference Level: 1.5 Volt

Output Load: 1 TTL Gate and  $C_L \sim 100pF$

## Capacitance<sup>[2]</sup> $T_A = 25^\circ C$ , $f = 1MHz$

Symbol	Test	Limits (pF)	
		Typ.	Max.
$C_{IN}$	Input Capacitance (All Input Pins) $V_{IN} = 0V$	4	8
$C_{OUT}$	Output Capacitance $V_{OUT} = 0V$	8	12

# Waveforms

**READ CYCLE**

**WRITE CYCLE**


- NOTES:
1. Typical values are for  $T_A = 25^\circ\text{C}$  and nominal supply voltage.
  2. This parameter is periodically sampled and is not 100% tested.
  3. OD may be tied low for separate I/O operation.
  4. During the write cycle, OD is "high" for common I/O and "don't care" for separate I/O operation.

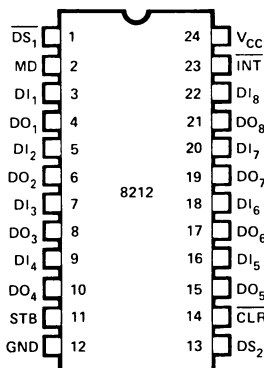
## EIGHT-BIT INPUT/OUTPUT PORT

- Fully Parallel 8-Bit Data Register and Buffer
- Service Request Flip-Flop for Interrupt Generation
- Low Input Load Current — .25 mA Max.
- Three State Outputs
- Outputs Sink 15 mA
- 3.65V Output High Voltage for Direct Interface to 8080 CPU or 8088 CPU
- Asynchronous Register Clear
- Replaces Buffers, Latches and Multiplexers in Microcomputer Systems
- Reduces System Package Count

The 8212 input/output port consists of an 8-bit latch with 3-state output buffers along with control and device selection logic. Also included is a service request flip-flop for the generation and control of interrupts to the microprocessor.

The device is multimode in nature. It can be used to implement latches, gated buffers or multiplexers. Thus, all of the principal peripheral and input/output functions of a microcomputer system can be implemented with this device.

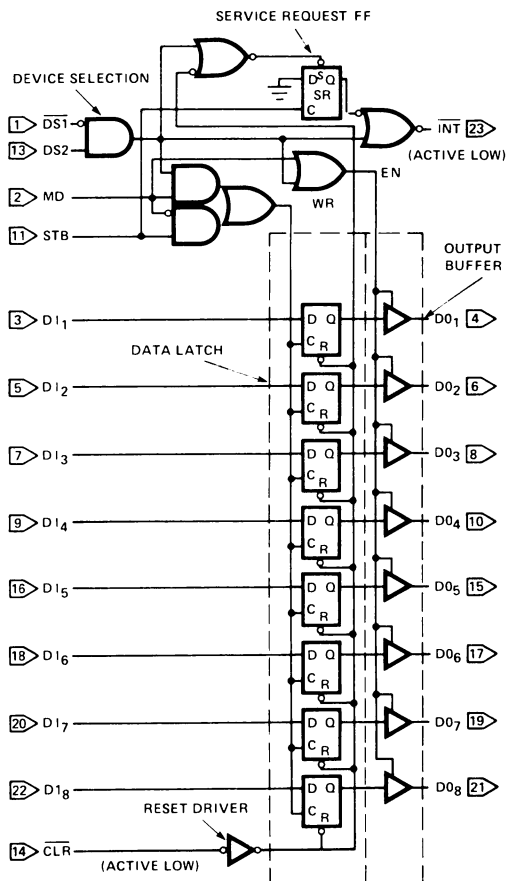
### PIN CONFIGURATION



### PIN NAMES

DI <sub>1</sub> -DI <sub>8</sub>	DATA IN
DO <sub>1</sub> -DO <sub>8</sub>	DATA OUT
DS <sub>1</sub> -DS <sub>2</sub>	DEVICE SELECT
MD	MODE
STB	STROBE
INT	INTERRUPT (ACTIVE LOW)
CLR	CLEAR (ACTIVE LOW)

### LOGIC DIAGRAM



## Functional Description

### Data Latch

The 8 flip-flops that make up the data latch are of a "D" type design. The output (Q) of the flip-flop will follow the data input (D) while the clock input (C) is high. Latching will occur when the clock (C) returns low.

The data latch is cleared by an asynchronous reset input (CLR). (Note: Clock (C) Overrides Reset (CLR).)

### Output Buffer

The outputs of the data latch (Q) are connected to 3-state, non-inverting output buffers. These buffers have a common control line (EN); this control line either enables the buffer to transmit the data from the outputs of the data latch (Q) or disables the buffer, forcing the output into a high impedance state. (3-state)

This high-impedance state allows the designer to connect the 8212 directly onto the microprocessor bi-directional data bus.

### Control Logic

The 8212 has control inputs  $\overline{DS1}$ , DS2, MD and STB. These inputs are used to control device selection, data latching, output buffer state and service request flip-flop.

### $\overline{DS1}$ , DS2 (Device Select)

These 2 inputs are used for device selection. When  $\overline{DS1}$  is low and DS2 is high ( $\overline{DS1} \cdot DS2$ ) the device is selected. In the selected state the output buffer is enabled and the service request flip-flop (SR) is asynchronously set.

### MD (Mode)

This input is used to control the state of the output buffer and to determine the source of the clock input (C) to the data latch.

When MD is high (output mode) the output buffers are enabled and the source of clock (C) to the data latch is from the device selection logic ( $\overline{DS1} \cdot DS2$ ). When MD is low (input mode) the output buffer state is determined by the device selection logic ( $\overline{DS1} \cdot DS2$ ) and the source of clock (C) to the data latch is the STB (Strobe) input.

### STB (Strobe)

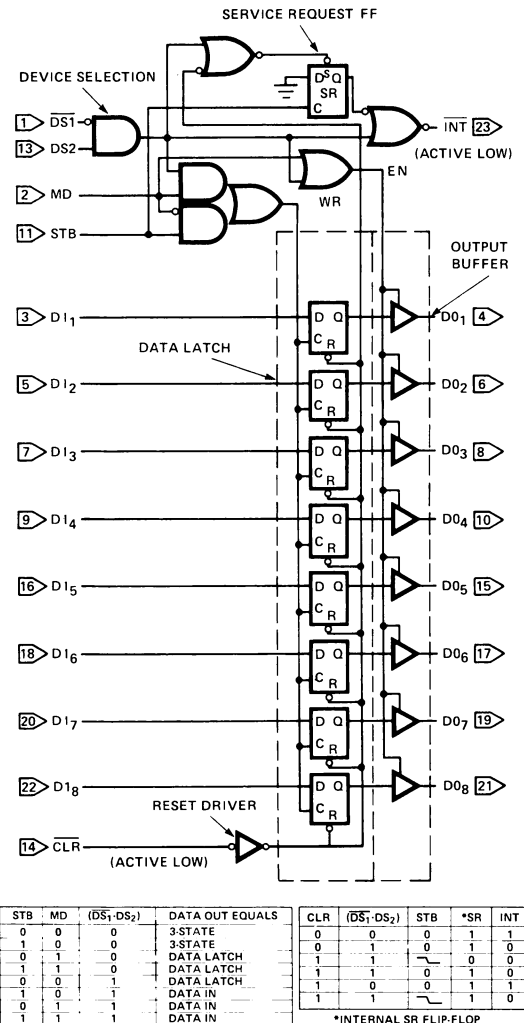
This input is used as the clock (C) to the data latch for the input mode MD = 0 and to synchronously reset the service request flip-flop (SR).

Note that the SR flip-flop is negative edge triggered.

### Service Request Flip-Flop

The (SR) flip-flop is used to generate and control interrupts in microcomputer systems. It is asynchronously set by the CLR input (active low). When the (SR) flip-flop is set it is in the non-interrupting state.

The output of the (SR) flip-flop (Q) is connected to an inverting input of a "NOR" gate. The other input to the "NOR" gate is non-inverting and is connected to the device selection logic ( $\overline{DS1} \cdot DS2$ ). The output of the "NOR" gate (INT) is active low (interrupting state) for connection to active low input priority generating circuits.



## Applications Of The 8212 -- For Microcomputer Systems

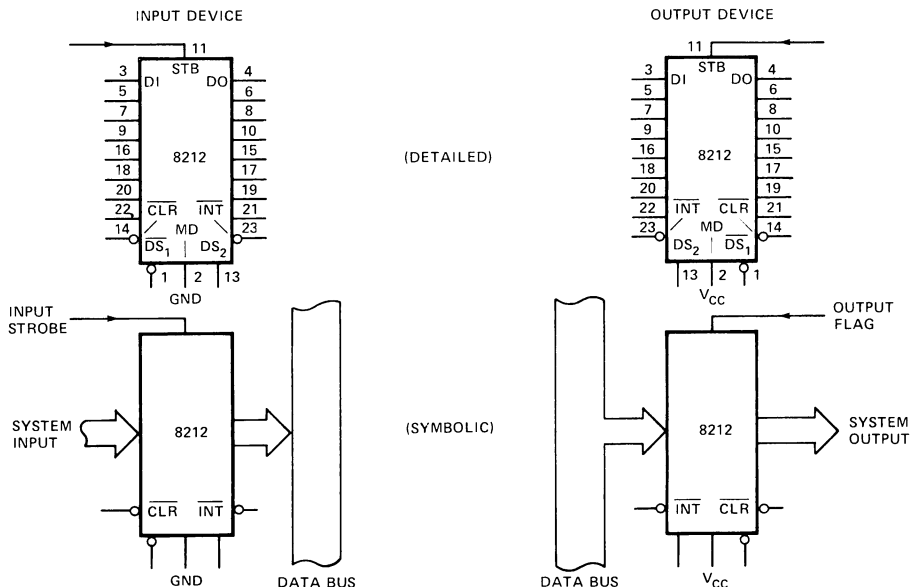
I	Basic Schematic Symbol	VII	8080 Status Latch
II	Gated Buffer	VIII	8008 System
III	Bi-Directional Bus Driver	IX	8080 System:
IV	Interrupting Input Port		8 Input Ports
V	Interrupt Instruction Port		8 Output Ports
VI	Output Port		8 Level Priority Interrupt

### I. Basic Schematic Symbols

Two examples of ways to draw the 8212 on system schematics—(1) the top being the detailed view showing pin numbers, and (2) the bottom being the symbolic view showing the system input or output

as a system bus (bus containing 8 parallel lines). The output to the data bus is symbolic in referencing 8 parallel lines.

#### BASIC SCHEMATIC SYMBOLS



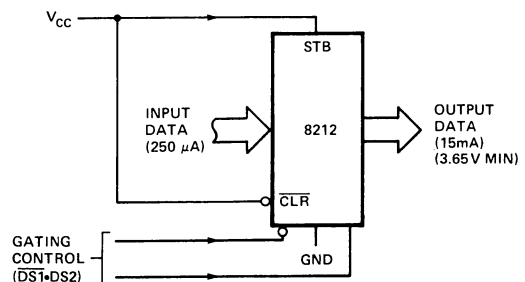
### II. Gated Buffer ( 3 - STATE )

The simplest use of the 8212 is that of a gated buffer. By tying the mode signal low and the strobe input high, the data latch is acting as a straight through gate. The output buffers are then enabled from the device selection logic  $\overline{DS1}$  and  $\overline{DS2}$ .

When the device selection logic is false, the outputs are 3-state.

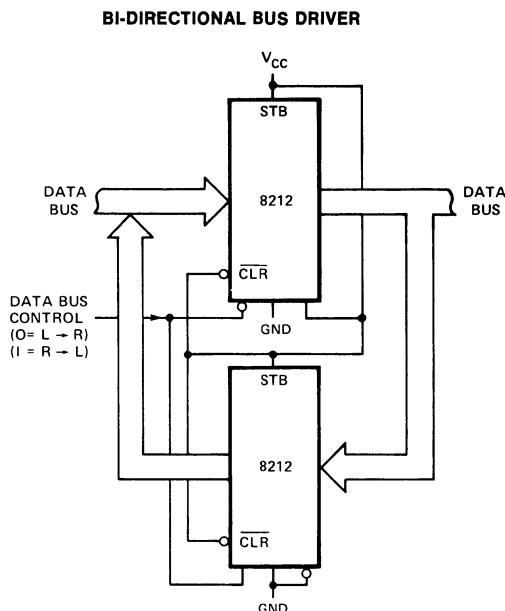
When the device selection logic is true, the input data from the system is directly transferred to the output. The input data load is 250 micro amps. The output data can sink 15 milli amps. The minimum high output is 3.65 volts.

#### GATED BUFFER 3-STATE



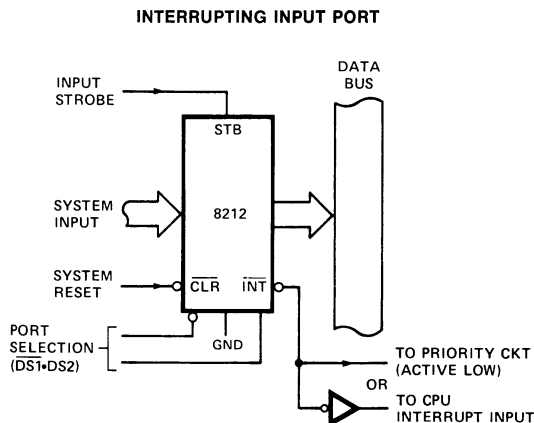
### III. Bi-Directional Bus Driver

A pair of 8212's wired (back-to-back) can be used as a symmetrical drive, bi-directional bus driver. The devices are controlled by the data bus input control which is connected to  $\overline{DS1}$  on the first 8212 and to  $\overline{DS2}$  on the second. One device is active, and acting as a straight through buffer the other is in 3-state mode. This is a very useful circuit in small system design.



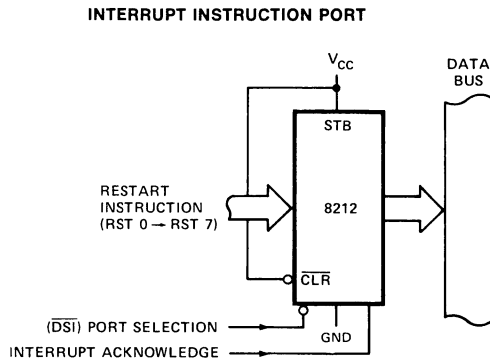
### IV. Interrupting Input Port

This use of an 8212 is that of a system input port that accepts a strobe from the system input source, which in turn clears the service request flip-flop and interrupts the processor. The processor then goes through a service routine, identifies the port, and causes the device selection logic to go true — enabling the system input data onto the data bus.



### V. Interrupt Instruction Port

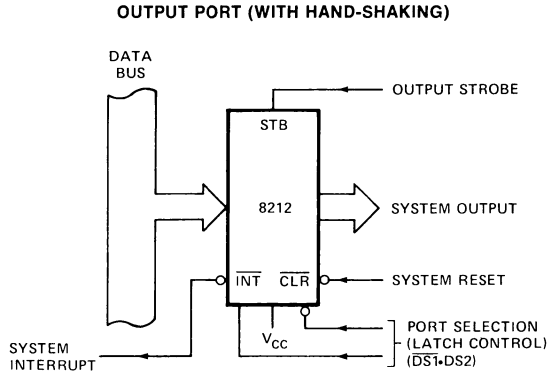
The 8212 can be used to gate the interrupt instruction, normally RESTART instructions, onto the data bus. The device is enabled from the interrupt acknowledge signal from the microprocessor and from a port selection signal. This signal is normally tied to ground. ( $\overline{DS1}$  could be used to multiplex a variety of interrupt instruction ports onto a common bus).





## VI. Output Port (With Hand-Shaking)

The 8212 can be used to transmit data from the data bus to a system output. The output strobe could be a hand-shaking signal such as "reception of data" from the device that the system is outputting to. It in turn, can interrupt the system signifying the reception of data. The selection of the port comes from the device selection logic. ( $\overline{DS1} \cdot \overline{DS2}$ )

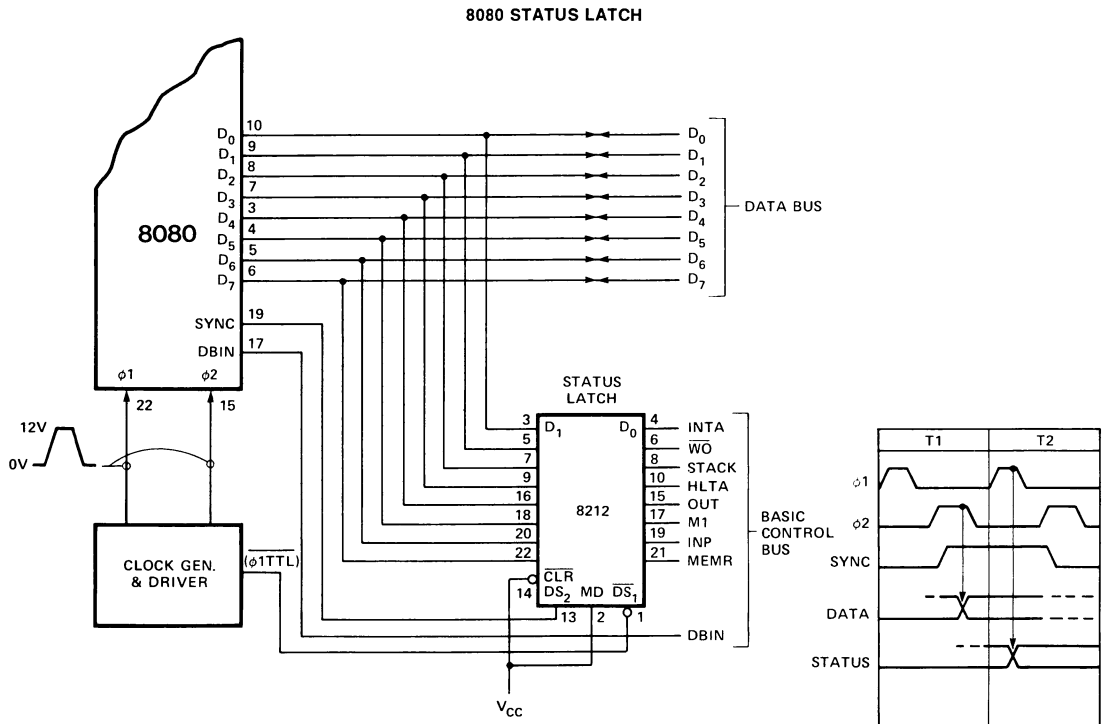


## VII. 8080 Status Latch

Here the 8212 is used as the status latch for an 8080 microcomputer system. The input to the 8212 latch is directly from the 8080 data bus. Timing shows that when the SYNC signal is true, which is connected to the DS2 input and the phase 1 signal is true, which is a TTL level coming from the clock generator; then, the status data will be latched into the 8212.

Note: The mode signal is tied high so that the output on the latch is active and enabled all the time.

It is shown that the two areas of concern are the bidirectional data bus of the microprocessor and the control bus.



## Absolute Maximum Ratings\*

Temperature Under Bias Plastic . . .  $-65^{\circ}\text{C}$  to  $+75^{\circ}\text{C}$

Storage Temperature . . . . .  $-65^{\circ}\text{C}$  to  $+160^{\circ}\text{C}$

All Output or Supply Voltages . . . .  $-0.5$  to  $+7$  Volts

All Input Voltages . . . . .  $-1.0$  to  $5.5$  Volts

Output Currents . . . . .  $125$  mA

\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied.

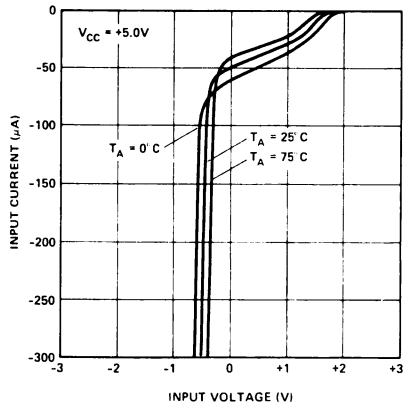
## D.C. Characteristics

$T_A = 0^{\circ}\text{C}$  to  $+75^{\circ}\text{C}$   $V_{CC} = +5\text{V} \pm 5\%$

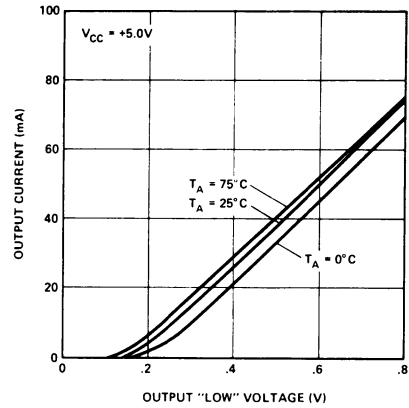
Symbol	Parameter	Limits			Unit	Test Conditions
		Min.	Typ.	Max.		
$I_F$	Input Load Current ACK, DS <sub>2</sub> , CR, DI <sub>1</sub> -DI <sub>8</sub> Inputs			$-.25$	mA	$V_F = .45\text{V}$
$I_F$	Input Load Current MD Input			$-.75$	mA	$V_F = .45\text{V}$
$I_F$	Input Load Current DS <sub>1</sub> Input			$-1.0$	mA	$V_F = .45\text{V}$
$I_R$	Input Leakage Current ACK, DS, CR, DI <sub>1</sub> -DI <sub>8</sub> Inputs			10	$\mu\text{A}$	$V_R = 5.25\text{V}$
$I_R$	Input Leakage Current MO Input			30	$\mu\text{A}$	$V_R = 5.25\text{V}$
$I_R$	Input Leakage Current DS <sub>1</sub> Input			40	$\mu\text{A}$	$V_R = 5.25\text{V}$
$V_C$	Input Forward Voltage Clamp			$-1$	V	$I_C = -5$ mA
$V_{IL}$	Input "Low" Voltage			.85	V	
$V_{IH}$	Input "High" Voltage	2.0			V	
$V_{OL}$	Output "Low" Voltage			.45	V	$I_{OL} = 15$ mA
$V_{OH}$	Output "High" Voltage	3.65	4.0		V	$I_{OH} = -1$ mA
$I_{SC}$	Short Circuit Output Current	$-15$		$-75$	mA	$V_O = 0$ V
$ I_O $	Output Leakage Current High Impedance State			20	$\mu\text{A}$	$V_O = .45\text{V}/5.25\text{V}$
$I_{CC}$	Power Supply Current		90	130	mA	

## Typical Characteristics

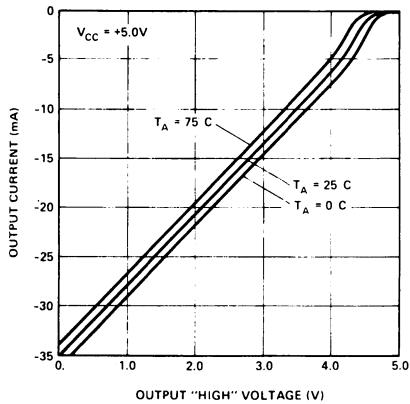
INPUT CURRENT VS. INPUT VOLTAGE



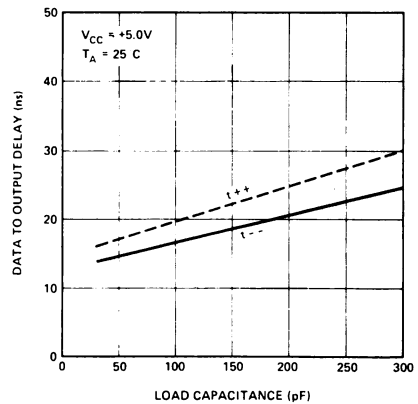
OUTPUT CURRENT VS. OUTPUT "LOW" VOLTAGE



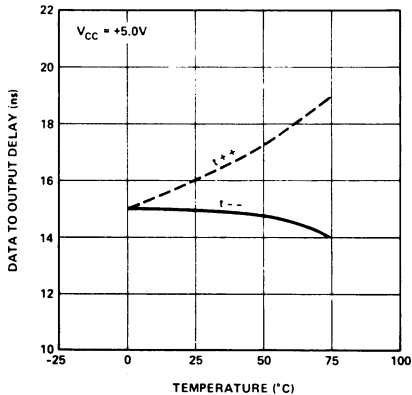
OUTPUT CURRENT VS. OUTPUT "HIGH" VOLTAGE



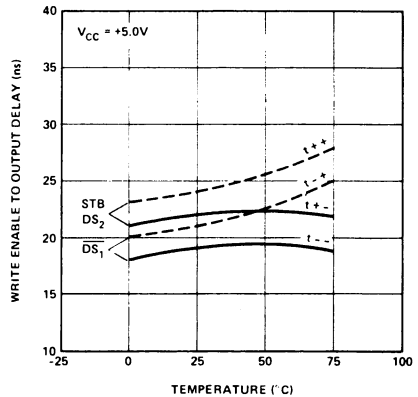
DATA TO OUTPUT DELAY VS. LOAD CAPACITANCE



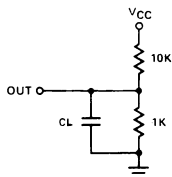
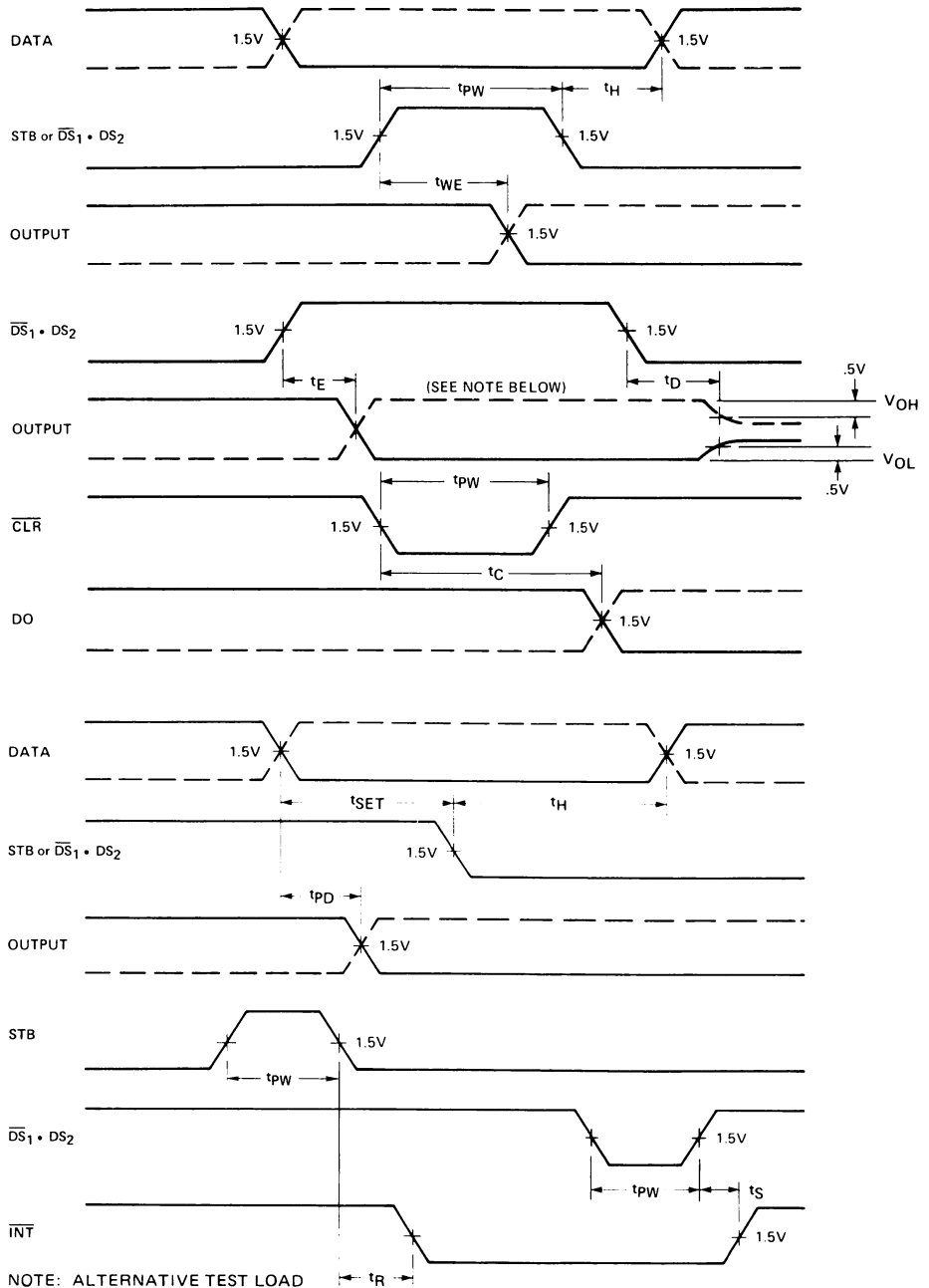
DATA TO OUTPUT DELAY VS. TEMPERATURE



WRITE ENABLE TO OUTPUT DELAY VS. TEMPERATURE



## Timing Diagram



## A.C. Characteristics

$T_A = 0^\circ\text{C}$  to  $+75^\circ\text{C}$      $V_{CC} = +5\text{V} \pm 5\%$

Symbol	Parameter	Limits			Unit	Test Conditions
		Min.	Typ.	Max.		
$t_{pw}$	Pulse Width	30			ns	
$t_{pd}$	Data To Output Delay			30	ns	
$t_{we}$	Write Enable To Output Delay			40	ns	
$t_{set}$	Data Setup Time	15			ns	
$t_h$	Data Hold Time	20			ns	
$t_r$	Reset To Output Delay			40	ns	
$t_s$	Set To Output Delay			30	ns	
$t_o$	Output Enable/Disable Time			45	ns	
$t_c$	Clear To Output Delay			55	ns	

CAPACITANCE \*     $F = 1\text{ MHz}$      $V_{BIAS} = 2.5\text{V}$      $V_{CC} = +5\text{V}$      $T_A = 25^\circ\text{C}$

Symbol	Test	LIMITS	
		Typ.	Max.
$C_{IN}$	$DS_1$ , MD Input Capacitance	9 pF	12 pF
$C_{IN}$	$DS_2$ , CK, ACK, $DI_1$ - $DI_8$ Input Capacitance	5 pF	9 pF
$C_{OUT}$	$DO_1$ - $DO_8$ Output Capacitance	8 pF	12 pF

\*This parameter is sampled and not 100% tested.

## Switching Characteristics

### CONDITIONS OF TEST

Input Pulse Amplitude = 2.5 V

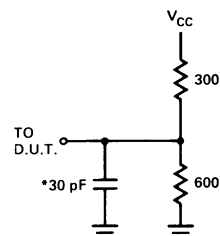
Input Rise and Fall Times 5 ns

Between 1V and 2V Measurements made at 1.5V

with 15 mA & 30 pF Test Load

### TEST LOAD

15 mA & 30 pF



\* INCLUDING JIG & PROBE CAPACITANCE



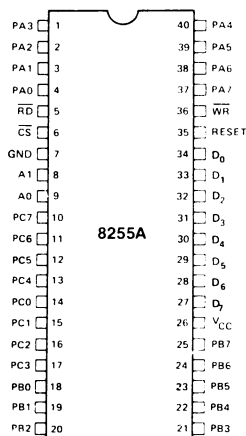
## 8255A PROGRAMMABLE PERIPHERAL INTERFACE

- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with MCS-80™ Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- 40 Pin Dual-In-Line Package
- Reduces System Package Count
- Improved DC Driving Capability

The 8255A is a general purpose programmable I/O device designed for use with both the 8008 and 8080 microprocessors. It has 24 I/O pins which may be individually programmed in two groups of twelve and used in three major modes of operation. In the first mode (Mode 0), each group of twelve I/O pins may be programmed in sets of 4 to be input or output. In Mode 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining four pins three are used for handshaking and interrupt control signals. The third mode of operation (Mode 2) is a Bi-directional Bus mode which uses 8 lines for a bi-directional bus, and five lines, borrowing one from the other group, for handshaking.

Other features of the 8255A include bit set and reset capability and the ability to source 1 mA of current at 1.5 volts. This allows darlington transistors to be directly driven for applications such as printers and high voltage displays.

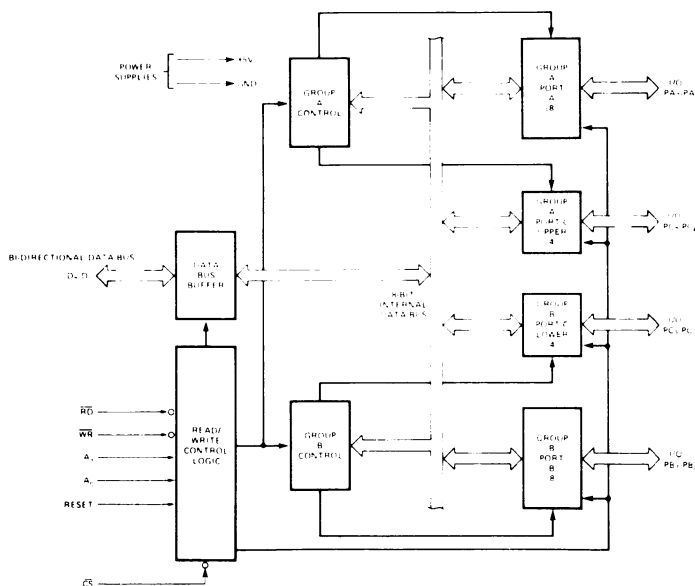
### PIN CONFIGURATION



### PIN NAMES

D <sub>7</sub> -D <sub>0</sub>	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
CS	CHIP SELECT
RD	READ INPUT
WR	WRITE INPUT
A <sub>0</sub> , A <sub>1</sub>	PORT ADDRESS
PA <sub>7</sub> -PA <sub>0</sub>	PORT A (BIT)
PB <sub>7</sub> -PB <sub>0</sub>	PORT B (BIT)
PC <sub>7</sub> -PC <sub>0</sub>	PORT C (BIT)
V <sub>CC</sub>	+5 VOLTS
GND	0 VOLTS

### 8255A BLOCK DIAGRAM



## 8255 BASIC FUNCTIONAL DESCRIPTION

### General

The 8255 is a Programmable Peripheral Interface (PPI) device designed for use in 8080 Microcomputer Systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the 8080 system bus. The functional configuration of the 8255 is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

### Data Bus Buffer

This 3-state, bi-directional, eight bit buffer is used to interface the 8255 to the 8080 system data bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions by the 8080 CPU. Control Words and Status information are also transferred through the Data Bus buffer.

### Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the 8080 CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

### ( $\overline{CS}$ )

**Chip Select:** A "low" on this input pin enables the communication between the 8255 and the 8080 CPU.

### ( $\overline{RD}$ )

**Read:** A "low" on this input pin enables the 8255 to send the Data or Status information to the 8080 CPU on the Data Bus. In essence, it allows the 8080 CPU to "read from" the 8255.

### ( $\overline{WR}$ )

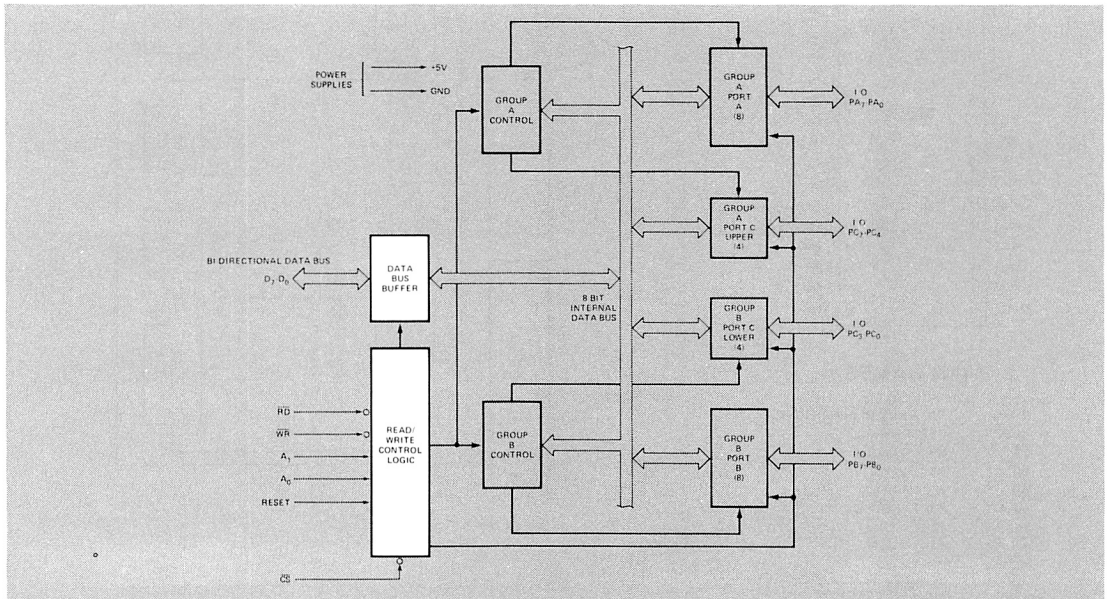
**Write:** A "low" on this input pin enables the 8080 CPU to write Data or Control words into the 8255.

### ( $A_0$ and $A_1$ )

**Port Select 0 and Port Select 1:** These input signals, in conjunction with the  $\overline{RD}$  and  $\overline{WR}$  inputs, control the selection of one of the three ports or the Control Word Register. They are normally connected to the least significant bits of the Address Bus ( $A_0$  and  $A_1$ ).

## 8255 BASIC OPERATION

$A_1$	$A_0$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A $\Rightarrow$ DATA BUS
0	1	0	1	0	PORT B $\Rightarrow$ DATA BUS
1	0	0	1	0	PORT C $\Rightarrow$ DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS $\Rightarrow$ PORT A
0	1	1	0	0	DATA BUS $\Rightarrow$ PORT B
1	0	1	0	0	DATA BUS $\Rightarrow$ PORT C
1	1	1	0	0	DATA BUS $\Rightarrow$ CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS $\Rightarrow$ 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS $\Rightarrow$ 3-STATE



8255 Block Diagram



**(RESET)**

**Reset:** A "high" on this input clears all internal registers including the Control Register and all ports (A, B, C) are set to the input mode.

**Group A and Group B Controls**

The functional configuration of each port is programmed by the systems software. In essence, the 8080 CPU "outputs" a control word to the 8255. The control word contains information such as "mode", "bit set", "bit reset" etc. that initializes the functional configuration of the 8255.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A — Port A and Port C upper (C7-C4)

Control Group B — Port B and Port C lower (C3-C0)

The Control Word Register can **Only** be written into. No Read operation of the Control Word Register is allowed.

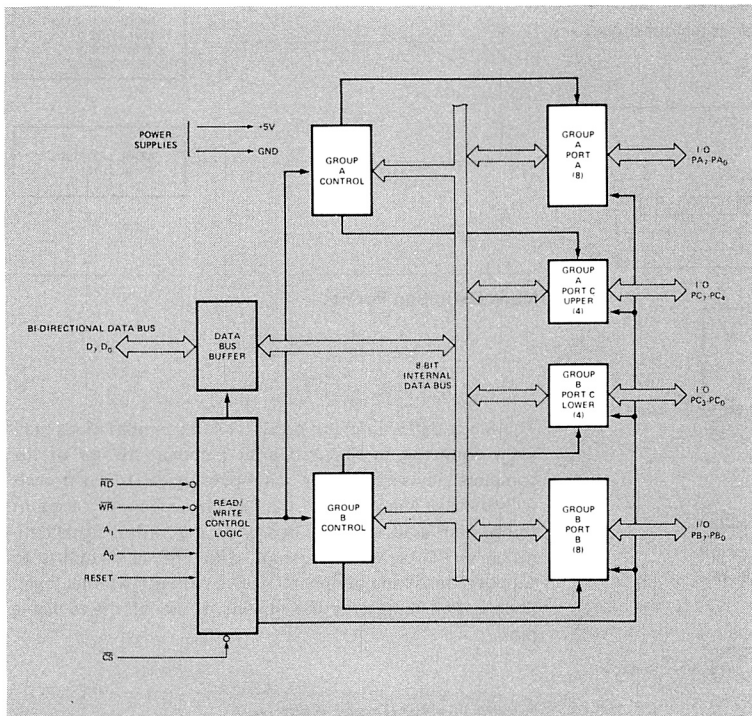
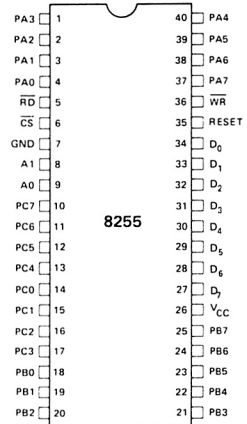
**Ports A, B, and C**

The 8255 contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.

**Port A:** One 8-bit data output latch/buffer and one 8-bit data input latch.

**Port B:** One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

**Port C:** One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with Ports A and B.

**8255 BLOCK DIAGRAM****PIN CONFIGURATION****PIN NAMES**

D7-D0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
CS	CHIP SELECT
RD	READ INPUT
WR	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
Vcc	+5 VOLTS
GND	0 VOLTS

## 8255 DETAILED OPERATIONAL DESCRIPTION

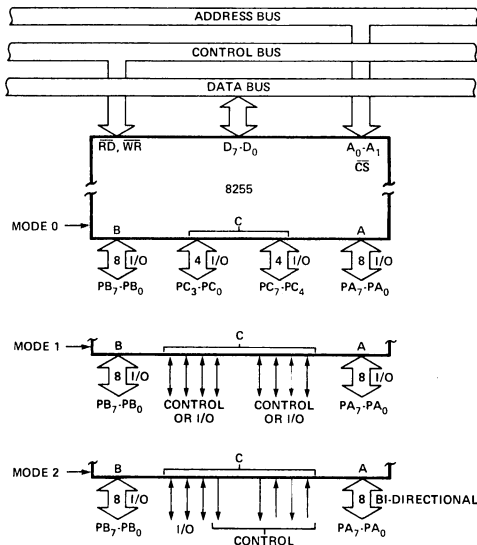
### Mode Selection

There are three basic modes of operation that can be selected by the system software:

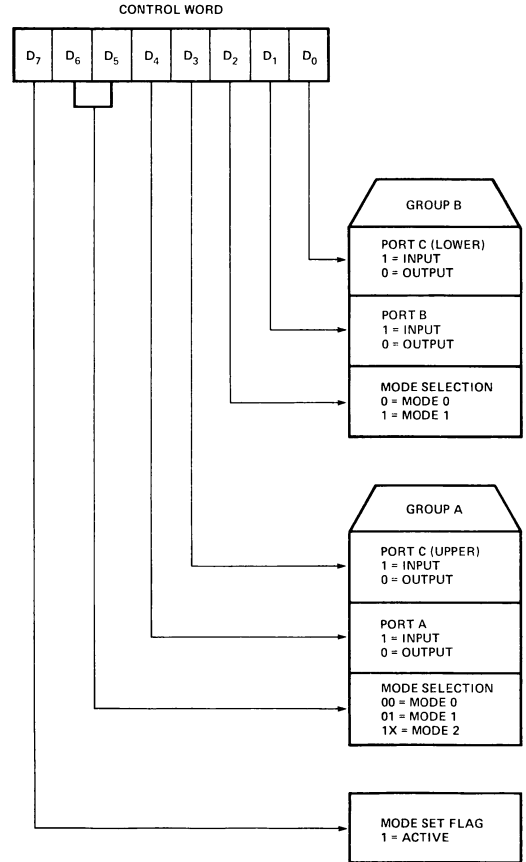
- Mode 0 – Basic Input/Output
- Mode 1 – Strobed Input/Output
- Mode 2 – Bi-Directional Bus

When the RESET input goes “high” all ports will be set to the Input mode (i.e., all 24 lines will be in the high impedance state). After the RESET is removed the 8255 can remain in the Input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single OUTput instruction. This allows a single 8255 to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be “tailored” to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.



Basic Mode Definitions and Bus Interface

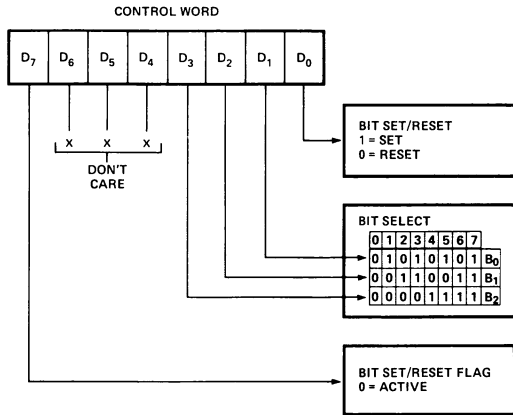


### Mode Definition Format

The Mode definitions and possible Mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255 has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

### Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.



When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

### Interrupt Control Functions

When the 8255 is programmed to operate in Mode 1 or Mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from Port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the Bit set/reset function of Port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) — INTE is SET — Interrupt enable

(BIT-RESET) — INTE is RESET — Interrupt disable

Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

### Bit Set/Reset Format

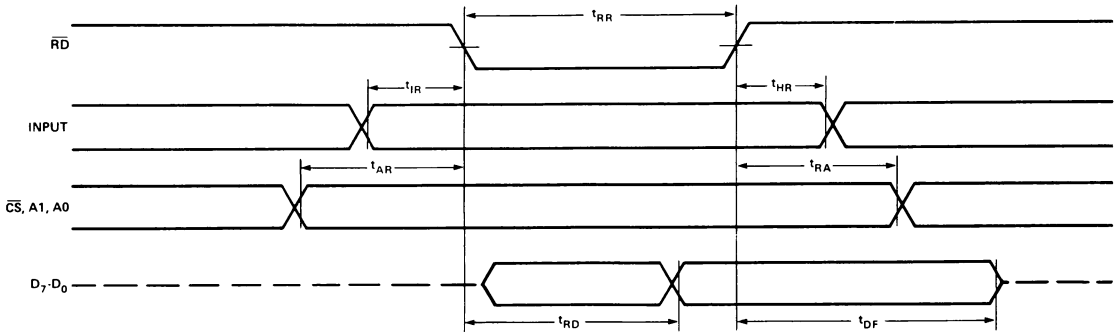
### Operating Modes

#### Mode 0 (Basic Input/Output)

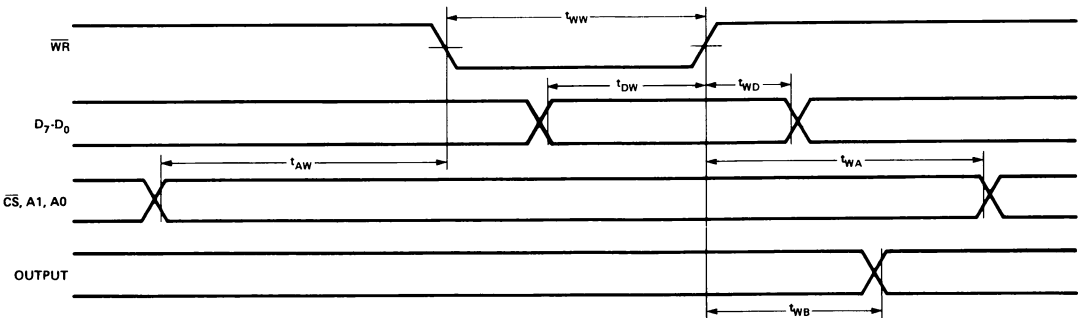
This functional configuration provides simple Input and Output operations for each of the three ports. No "hand-shaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.



#### Mode 0 (Basic Input)



#### Mode 0 (Basic Output)

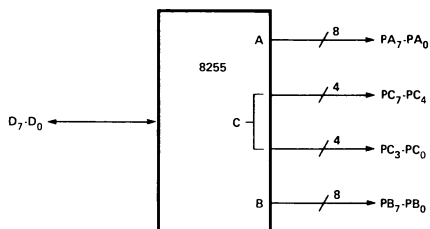
## MODE 0 PORT DEFINITION CHART

A		B		GROUP A			GROUP B	
D <sub>4</sub>	D <sub>3</sub>	D <sub>1</sub>	D <sub>0</sub>	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

## MODE 0 CONFIGURATIONS

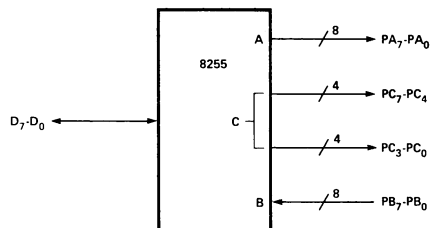
CONTROL WORD #0

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	0	0	0	0



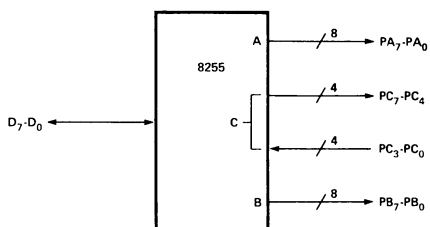
CONTROL WORD #2

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	0	0	1	0



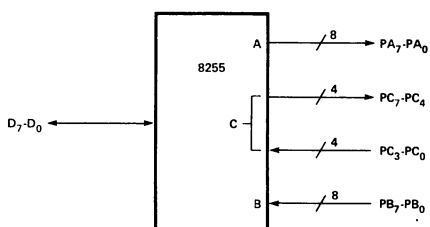
CONTROL WORD #1

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	0	0	0	1



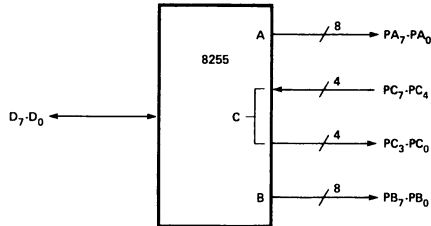
CONTROL WORD #3

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	0	0	1	1



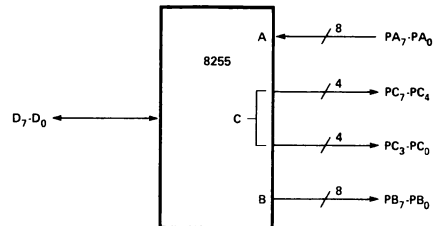
CONTROL WORD #4

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	0	0	0



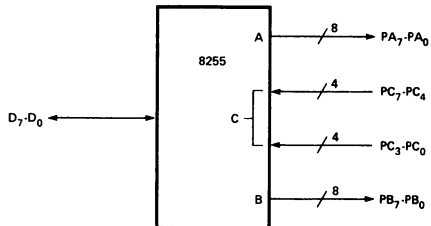
CONTROL WORD #8

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	0	0	0	0



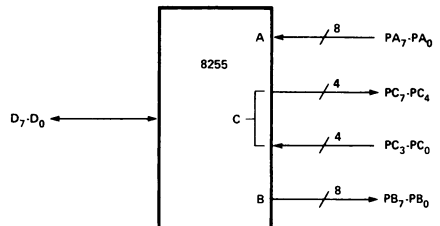
CONTROL WORD #5

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	0	0	1



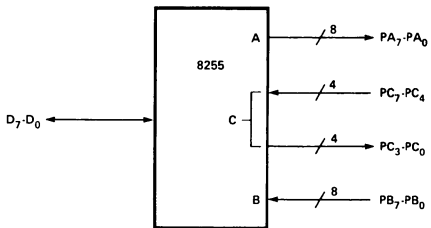
CONTROL WORD #9

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	0	0	0	1



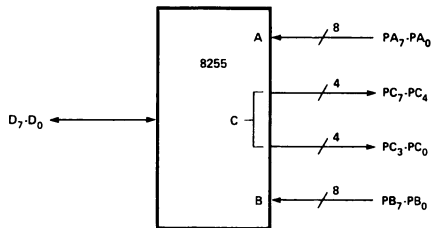
CONTROL WORD #6

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	0	1	0



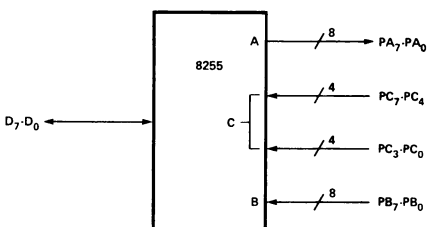
CONTROL WORD #10

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	0	0	1	0



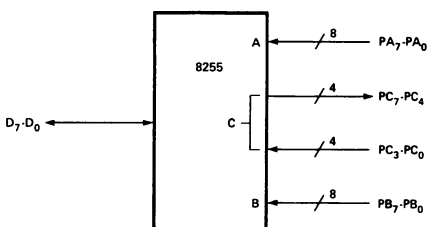
CONTROL WORD #7

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	0	1	1



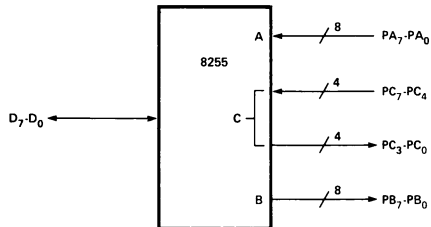
CONTROL WORD #11

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	0	0	1	1



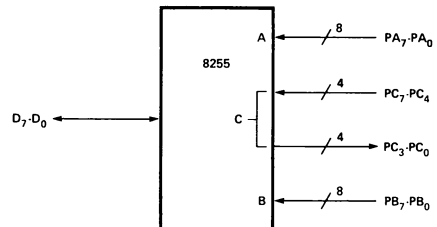
CONTROL WORD #12

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	1	0	0	0



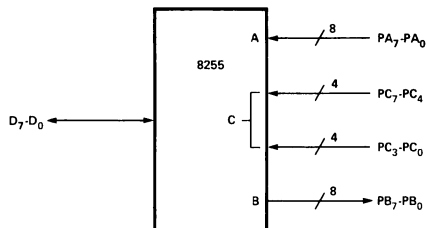
CONTROL WORD #14

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	1	0	1	0



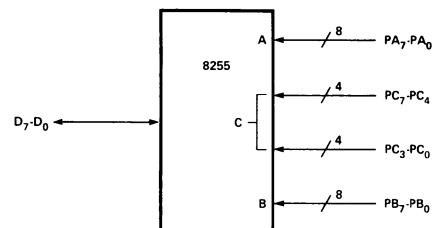
CONTROL WORD #13

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	1	0	0	1



CONTROL WORD #15

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	1	1	0	1	1



## Operating Modes

### Mode 1 (Strobed Input/Output)

This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In Mode 1, Port A and Port B use the lines on Port C to generate or accept these "handshaking" signals.

#### Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

## Input Control Signal Definition

### $\overline{STB}$ (Strobe Input)

A "low" on this input loads data into the input latch.

### IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by  $\overline{STB}$  input being low and is reset by the rising edge of the  $\overline{RD}$  input.

### INTR (Interrupt Request)

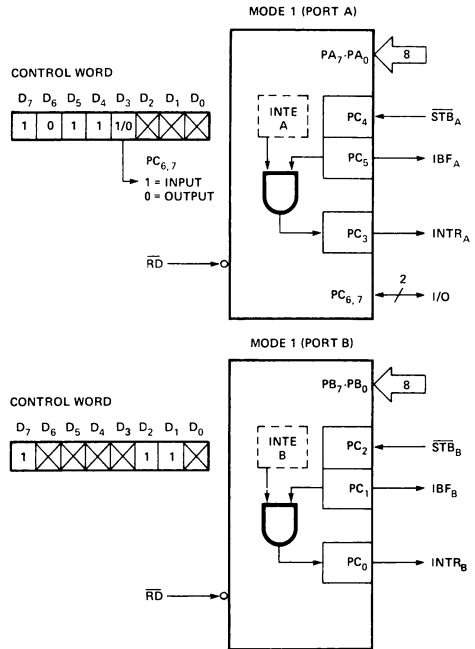
A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the  $\overline{STB}$  is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of  $\overline{RD}$ . This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

#### INTE A

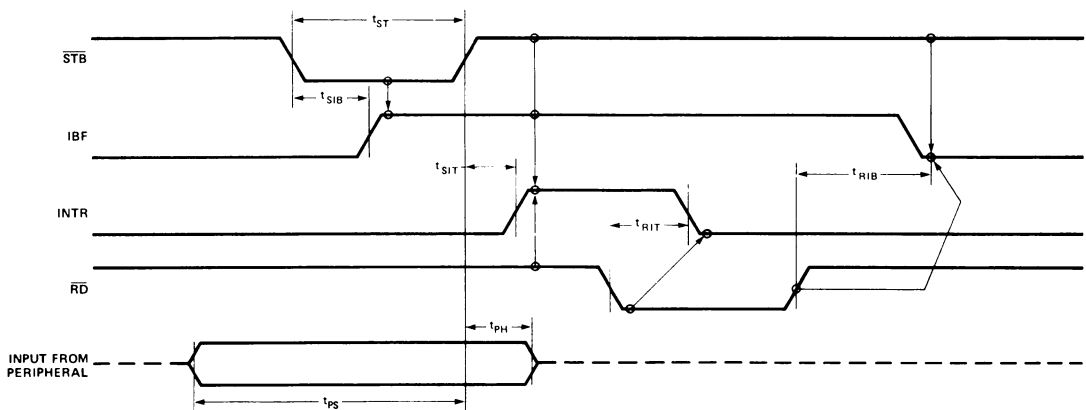
Controlled by bit set/reset of  $PC_4$ .

#### INTE B

Controlled by bit set/reset of  $PC_2$ .



## Mode 1 Input



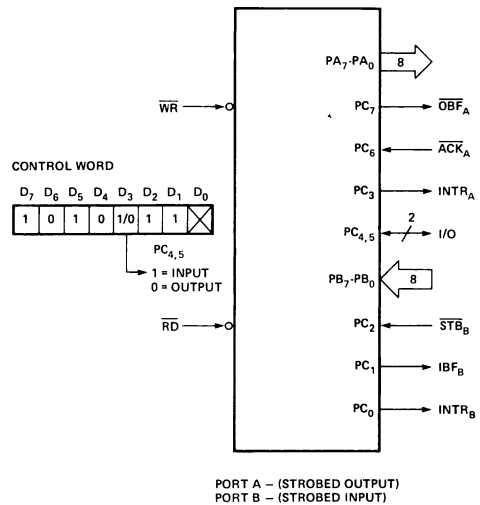
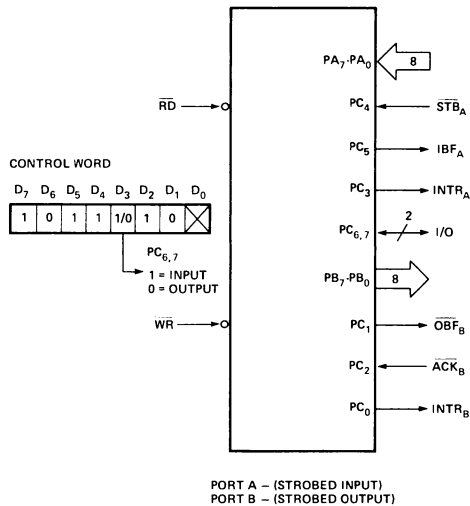
## Mode 1 (Strobed Input)





## Combinations of Mode 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.



## Operating Modes

### Mode 2 (Strobed Bi-Directional Bus I/O)

This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bi-directional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to Mode 1. Interrupt generation and enable/disable functions are also available.

Mode 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

### Bi-Directional Bus I/O Control Signal Definition

#### INTR (Interrupt Request)

A high on this output can be used to interrupt the CPU for both input or output operations.

## Output Operations

### $\overline{OBF}$ (Output Buffer Full)

The  $\overline{OBF}$  output will go "low" to indicate that the CPU has written data out to Port A.

### $\overline{ACK}$ (Acknowledge)

A "low" on this input enables the tri-state output buffer of Port A to send out the data. Otherwise, the output buffer will be in the high-impedance state.

### INTE 1 (The INTE Flip-Flop associated with $\overline{OBF}$ )

Controlled by bit set/reset of PC<sub>6</sub>.

## Input Operations

### $\overline{STB}$ (Strobe Input)

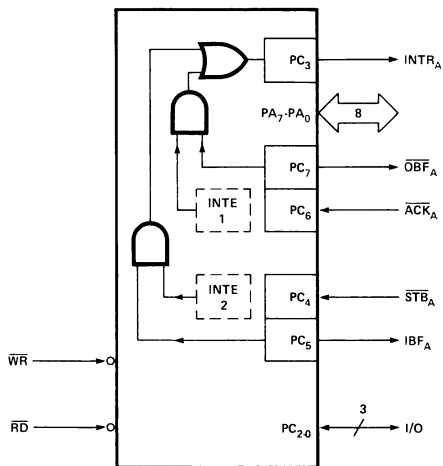
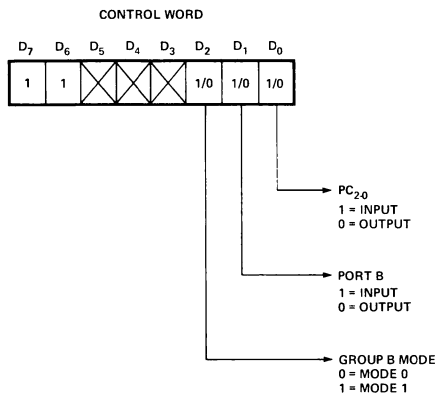
A "low" on this input loads data into the input latch.

### IBF (Input Buffer Full F/F)

A "high" on this output indicates that data has been loaded into the input latch.

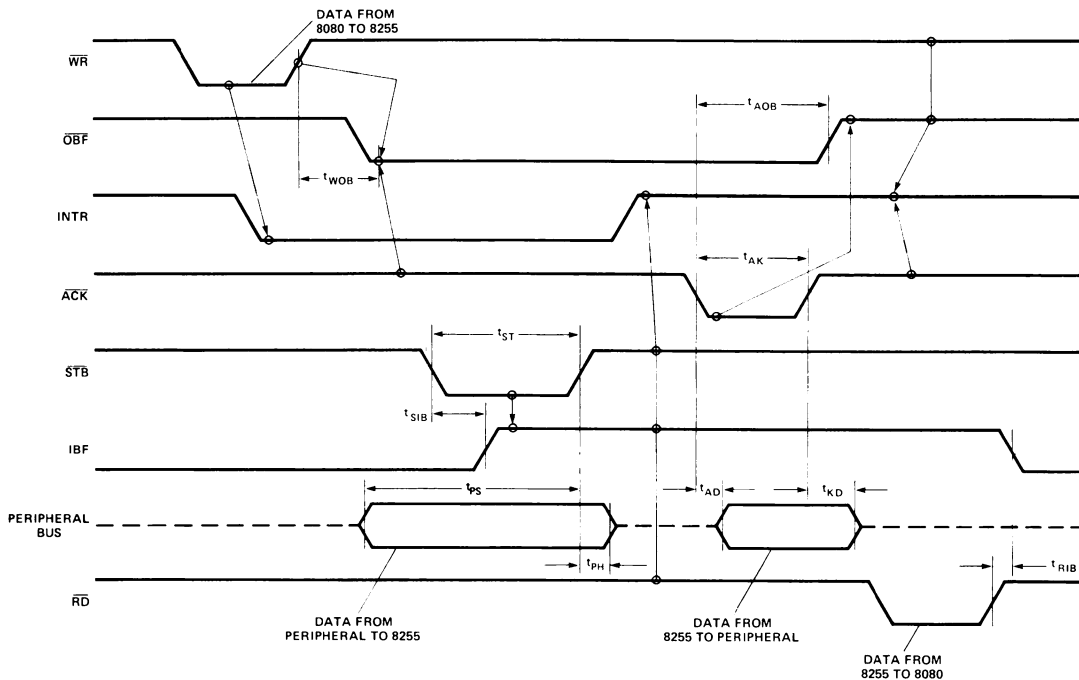
### INTE 2 (The INTE Flip-Flop associated with IBF)

Controlled by bit set/reset of PC<sub>4</sub>.



## Mode 2 Control Word

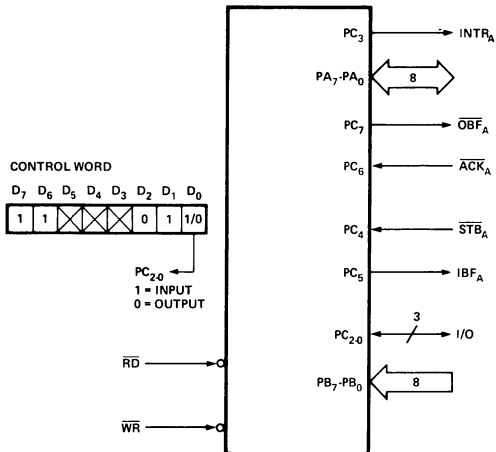
## Mode 2



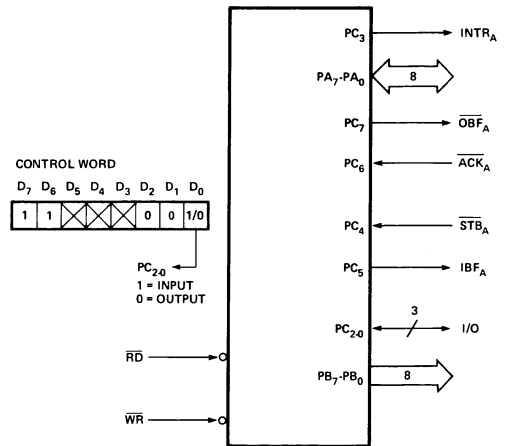
## Mode 2 (Bi-directional)

NOTE: Any sequence where  $\overline{WR}$  occurs before  $\overline{ACK}$  and  $\overline{STB}$  occurs before  $\overline{RD}$  is permissible.  
 (INTR = IBF • MASK •  $\overline{STB}$  •  $\overline{RD}$  + OBF • MASK • ACK •  $\overline{WR}$ )

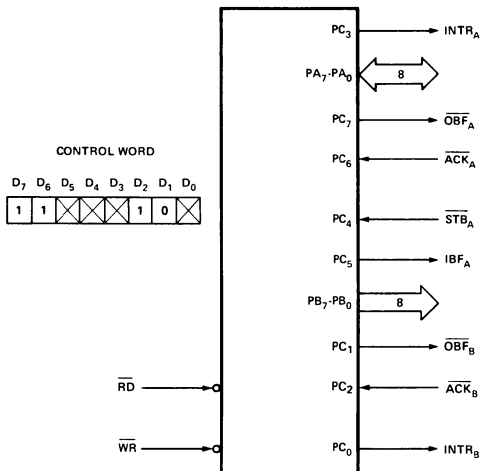
MODE 2 AND MODE 0 (INPUT)



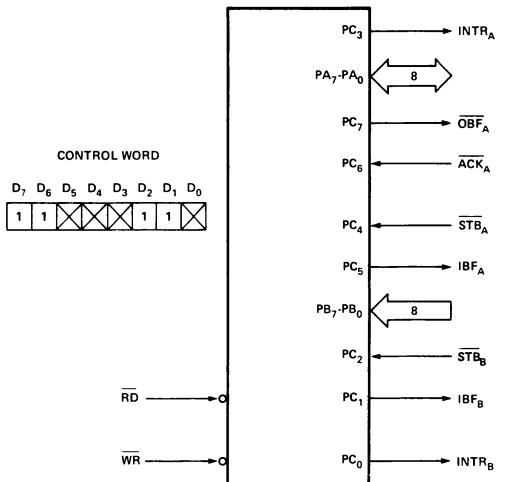
MODE 2 AND MODE 0 (OUTPUT)



MODE 2 AND MODE 1 (OUTPUT)



MODE 2 AND MODE 1 (INPUT)



MODE DEFINITION SUMMARY TABLE

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA <sub>0</sub>	IN	OUT	IN	OUT		
PA <sub>1</sub>	IN	OUT	IN	OUT		
PA <sub>2</sub>	IN	OUT	IN	OUT		
PA <sub>3</sub>	IN	OUT	IN	OUT		
PA <sub>4</sub>	IN	OUT	IN	OUT		
PA <sub>5</sub>	IN	OUT	IN	OUT		
PA <sub>6</sub>	IN	OUT	IN	OUT		
PA <sub>7</sub>	IN	OUT	IN	OUT		
PB <sub>0</sub>	IN	OUT	IN	OUT		
PB <sub>1</sub>	IN	OUT	IN	OUT		
PB <sub>2</sub>	IN	OUT	IN	OUT		
PB <sub>3</sub>	IN	OUT	IN	OUT		
PB <sub>4</sub>	IN	OUT	IN	OUT		
PB <sub>5</sub>	IN	OUT	IN	OUT		
PB <sub>6</sub>	IN	OUT	IN	OUT		
PB <sub>7</sub>	IN	OUT	IN	OUT		
PC <sub>0</sub>	IN	OUT	INTR <sub>B</sub>	INTR <sub>B</sub>	I/O I/O I/O INTR <sub>A</sub> STB <sub>A</sub> IBF <sub>A</sub> ACK <sub>A</sub> OBF <sub>A</sub>	
PC <sub>1</sub>	IN	OUT	IBF <sub>B</sub>	OBF <sub>B</sub>		
PC <sub>2</sub>	IN	OUT	STB <sub>B</sub>	ACK <sub>B</sub>		
PC <sub>3</sub>	IN	OUT	INTR <sub>A</sub>	INTR <sub>A</sub>		
PC <sub>4</sub>	IN	OUT	STB <sub>A</sub>	I/O		
PC <sub>5</sub>	IN	OUT	IBF <sub>A</sub>	I/O		
PC <sub>6</sub>	IN	OUT	I/O	ACK <sub>A</sub>		
PC <sub>7</sub>	IN	OUT	I/O	OBF <sub>A</sub>		

MODE 0  
OR MODE 1  
ONLY

### Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

If Programmed as Inputs —

All input lines can be accessed during a normal Port C read.

If Programmed as Outputs —

Bits in C upper (PC<sub>7</sub>-PC<sub>4</sub>) must be individually accessed using the bit set/reset function.

Bits in C lower (PC<sub>3</sub>-PC<sub>0</sub>) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

### Source Current Capability on Port B and Port C

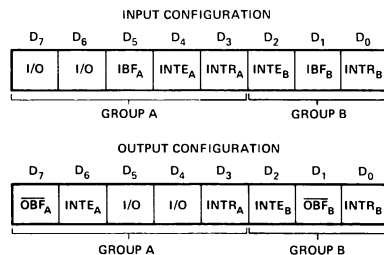
Any set of **eight** output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

### Reading Port C Status

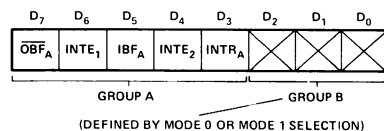
In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts “hand-shaking” signals with the peripheral device. Reading the contents of Port C

allows the programmer to test or verify the “status” of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.



### Mode 1 Status Word Format

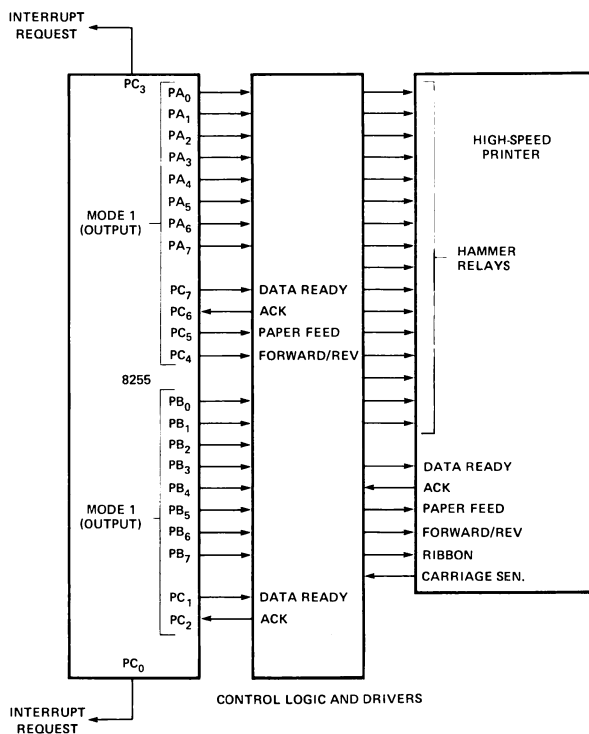


### Mode 2 Status Word Format

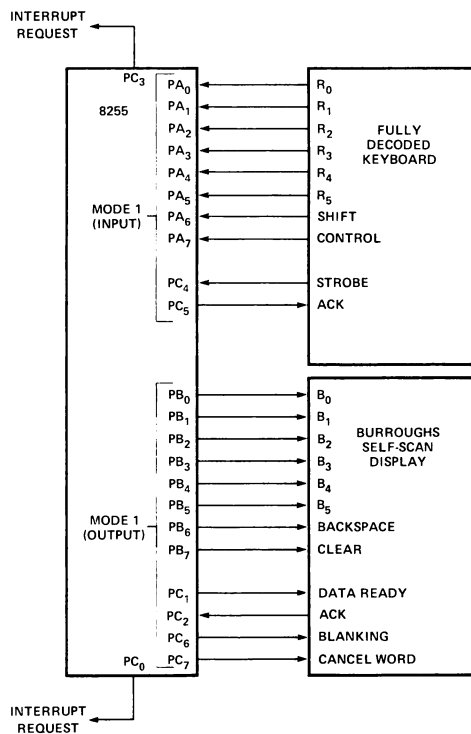
## APPLICATIONS OF THE 8255

The 8255 is a very powerful tool for interfacing peripheral equipment to the 8080 microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

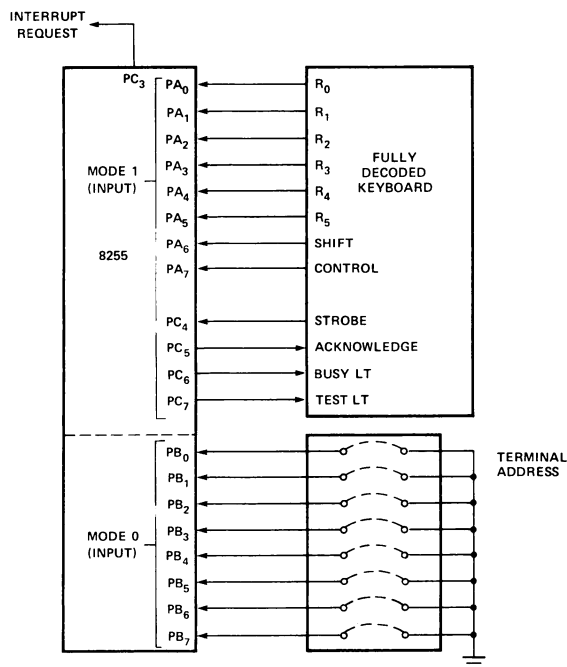
Each peripheral device in a Microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255 is programmed by the I/O service routine and becomes an extension of the systems software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the Detailed Operational Description, a control word can easily be developed to initialize the 8255 to exactly "fit" the application. Here are a few examples of typical applications of the 8255.



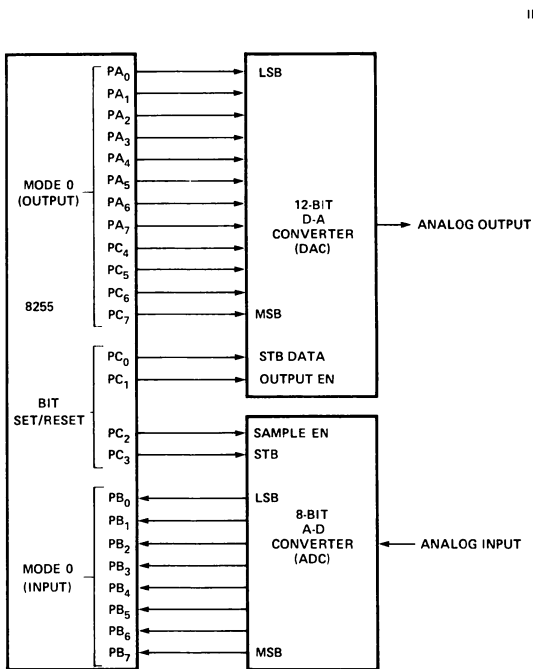
Printer Interface



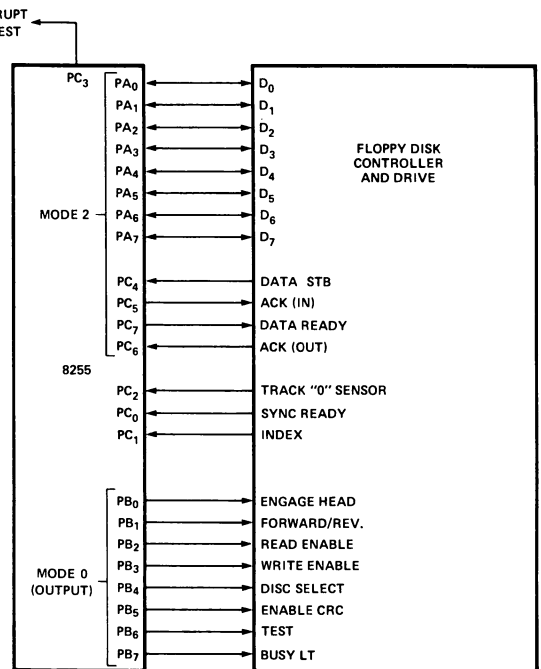
Keyboard and Display Interface



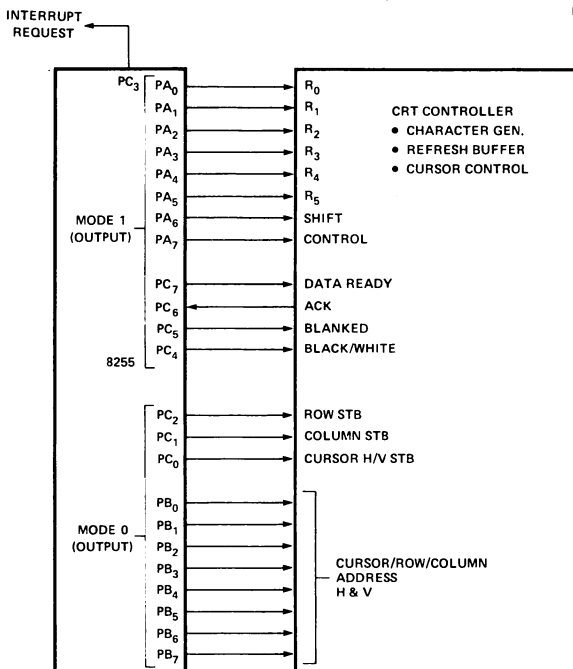
Keyboard and Terminal Address Interface



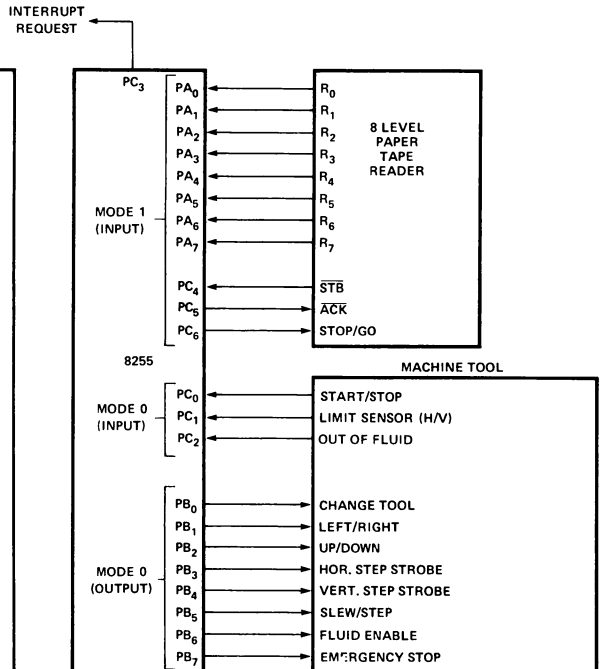
Digital to Analog, Analog to Digital



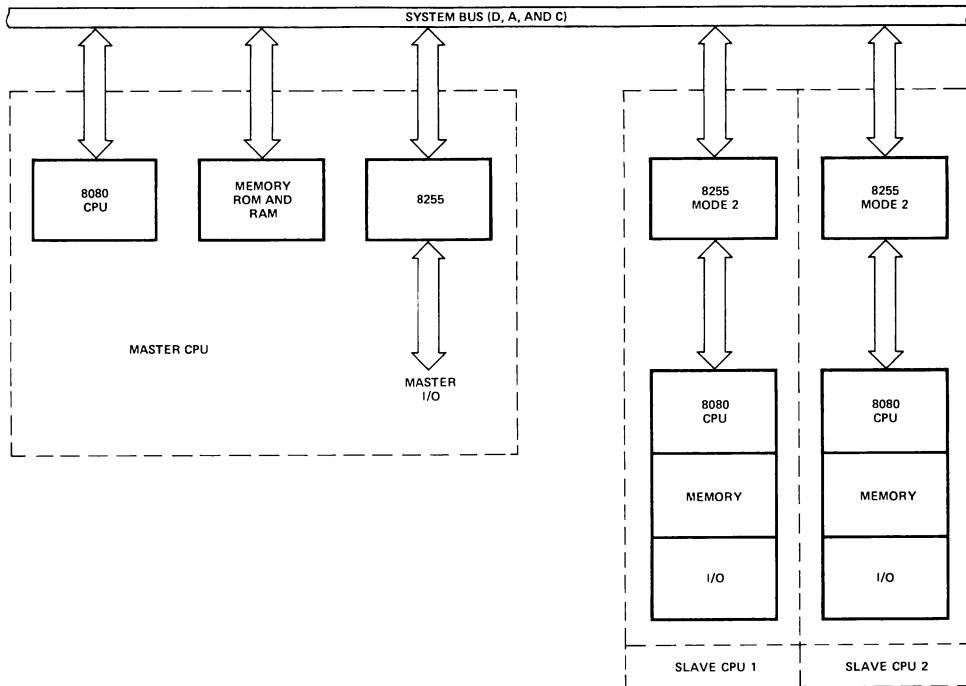
Basic Floppy Disc Interface



Basic CRT Controller Interface



Machine Tool Controller Interface



Distributed Intelligence Multi-Processor Interface

**ABSOLUTE MAXIMUM RATINGS\***

Ambient Temperature Under Bias. . . . . 0°C to 70°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage on Any Pin  
     With Respect to Ground. . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

*\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

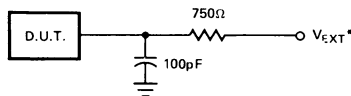
**D.C. CHARACTERISTICS**  $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ ,  $V_{CC} = +5V \pm 5\%$ ; GND = 0V

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$V_{IL}$	Input Low Voltage	-0.5	0.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC}$	V	
$I_{OL}(DB)$	Output Low Current (Data Bus)	2.5		mA	$V_{OL} = 0.45V$
$I_{OL}(PER)$	Output Low Current (Peripheral Port)	1.7		mA	$V_{OL} = 0.45V$
$I_{OH}(DB)$	Output High Current (Data Bus)	-400		$\mu A$	$V_{OH} = 2.4V$
$I_{OH}(PER)$	Output High Current (Peripheral Port)	-200		$\mu A$	$V_{OH} = 2.4V$
$I_{DAR}^{[1]}$	Darlington Drive Current	-1.0	-4.0	mA	$R_{EXT} = 750\Omega$ ; $V_{EXT} = 1.5V$
$I_{CC}$	Power Supply Current		120	mA	
$I_{IL}$	Input Leakage		10	$\mu A$	$V_{IN} = V_{CC}$
$I_{OFL}$	Output Float Leakage		10	$\mu A$	$V_{OUT} = GND + 0.45, V_{CC}$

Note: 1. Adaptable on any 8 pins from Ports Band C.

**CAPACITANCE**  $T_A = 25^\circ\text{C}$ ;  $V_{CC} = GND = 0V$ 

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT	TEST CONDITIONS
$C_{IN}$	Input Capacitance			10	pF	$f_c = 1\text{MHz}$
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to GND

**TEST LOAD CIRCUIT (FOR DB)**

\*  $V_{EXT}$  IS SET AT VARIOUS VOLTAGES DURING TESTING TO GUARANTEE THE SPECIFICATION.



**A.C. CHARACTERISTICS**  $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ ;  $V_{CC} = +5\text{V} \pm 5\%$ ;  $\text{GND} = 0\text{V}$ **BUS PARAMETERS:****READ:**

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AR}$	Address Stable Before $\overline{\text{READ}}$	0		ns	
$t_{RA}$	Address Stable After $\overline{\text{READ}}$	0		ns	
$t_{RR}$	$\overline{\text{READ}}$ Pulse Width	300		ns	
$t_{RD}$	Data Valid From $\overline{\text{READ}}$		250	ns	$\text{CL} = 100\text{ pF}$
$t_{DF}$	Data Float After $\overline{\text{READ}}$	10	150	ns	$\text{CL} = 100\text{ pF}$ $\text{CL} = 15\text{ pF}$
$t_{RV}$	Time Between $\overline{\text{READS}}$ and/or $\overline{\text{WRITES}}$	850		ns	

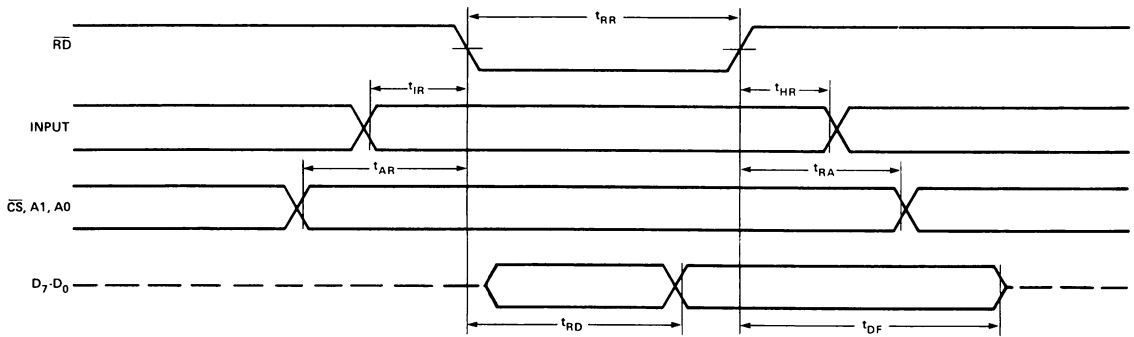
**WRITE:**

$t_{AW}$	Address Stable Before $\overline{\text{WRITE}}$	0		ns	
$t_{WA}$	Address Stable After $\overline{\text{WRITE}}$	20		ns	
$t_{WW}$	$\overline{\text{WRITE}}$ Pulse Width	400		ns	
$t_{DW}$	Data Valid To $\overline{\text{WRITE}}$ (T.E.)	100		ns	
$t_{WD}$	Data Valid After $\overline{\text{WRITE}}$	30		ns	

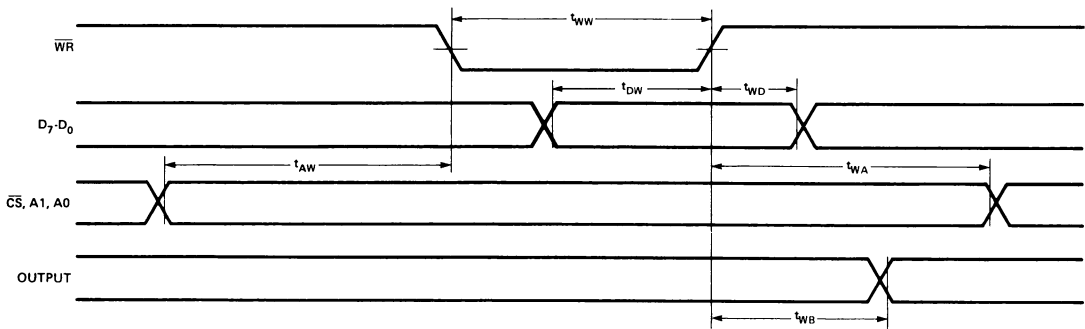
**OTHER TIMINGS:**

$t_{WB}$	$\overline{\text{WR}}=1$ To Output		350	ns	$\text{CL} = 100\text{ pF}$
$t_{IR}$	Peripheral Data Before $\overline{\text{RD}}$	0		ns	
$t_{HR}$	Peripheral Data After $\overline{\text{RD}}$	0		ns	
$t_{AK}$	$\overline{\text{ACK}}$ Pulse Width	300		ns	
$t_{ST}$	$\overline{\text{STB}}$ Pulse Width	500		ns	
$t_{PS}$	Per. Data Before T.E. Of $\overline{\text{STB}}$	0		ns	
$t_{PH}$	Per. Data After T.E. Of $\overline{\text{STB}}$	180		ns	
$t_{AD}$	$\overline{\text{ACK}}=0$ To Output		400	ns	$\text{CL} = 100\text{ pF}$
$t_{KD}$	$\overline{\text{ACK}}=1$ To Output Float	20	250	ns	$\text{CL} = 100\text{ pF}$ $\text{CL} = 15\text{ pF}$
$t_{WOB}$	$\overline{\text{WR}}=1$ To $\overline{\text{OBF}}=0$		650	ns	$\text{CL} = 100\text{ pF}$
$t_{AOB}$	$\overline{\text{ACK}}=0$ To $\overline{\text{OBF}}=1$		350	ns	$\text{CL} = 100\text{ pF}$
$t_{SIB}$	$\overline{\text{STB}}=0$ To $\text{IBF}=1$		300	ns	$\text{CL} = 100\text{ pF}$
$t_{RIB}$	$\overline{\text{RD}}=1$ To $\text{IBF}=0$		300	ns	$\text{CL} = 100\text{ pF}$
$t_{RIT}$	$\overline{\text{RD}}=0$ To $\text{INTR}=0$		400	ns	$\text{CL} = 100\text{ pF}$
$t_{SIT}$	$\overline{\text{STB}}=1$ To $\text{INTR}=1$		300	ns	$\text{CL} = 100\text{ pF}$
$t_{AIT}$	$\overline{\text{ACK}}=1$ To $\text{INTR}=1$		350	ns	$\text{CL} = 100\text{ pF}$
$t_{WIT}$	$\overline{\text{WR}}=0$ To $\text{INTR}=0$		850	ns	$\text{CL} = 100\text{ pF}$

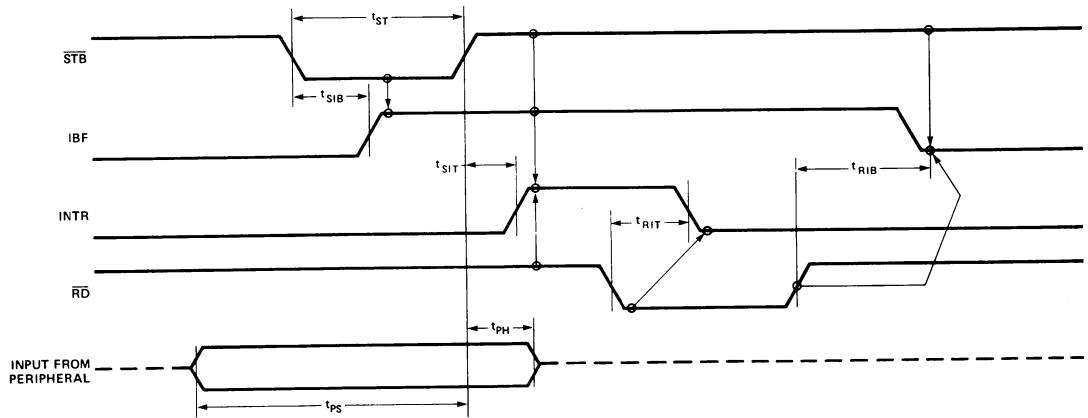
Note: Period of Reset pulse must be at least 50 $\mu\text{s}$  during or after power on.  
Subsequent Reset pulse can be 500 ns min.



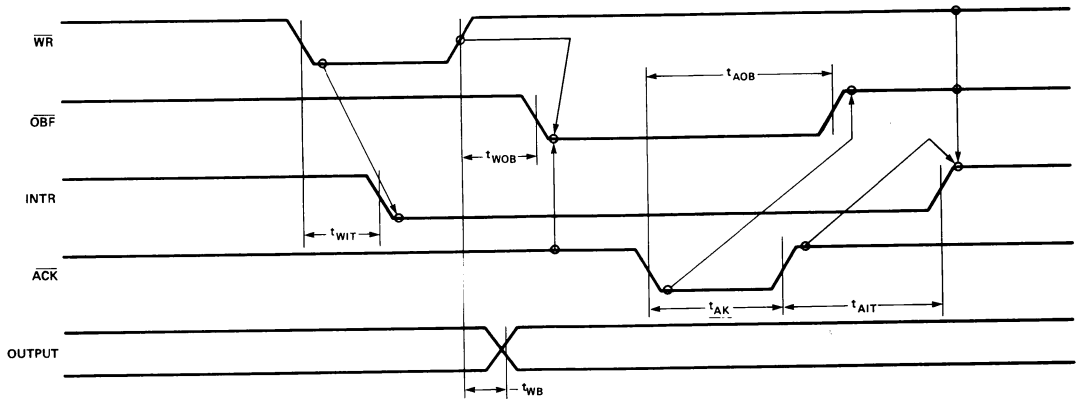
### Mode 0 (Basic Input)



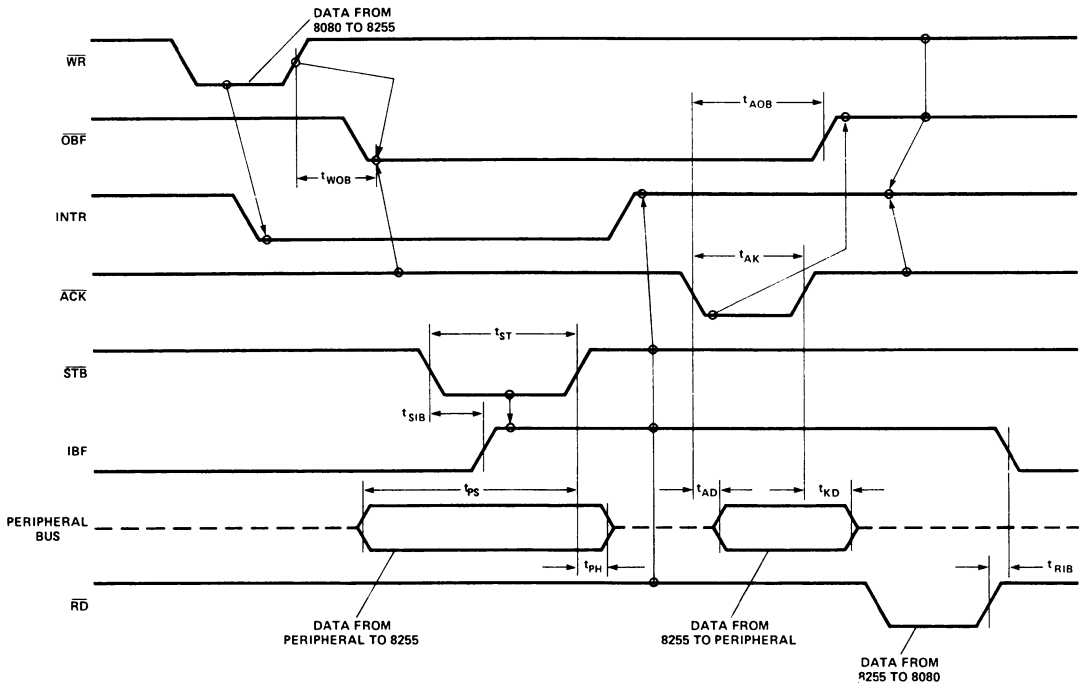
### Mode 0 (Basic Output)



### Mode 1 (Strobed Input)



### Mode 1 (Strobed Output)



### Mode 2 (Bi-directional)

NOTE: Any sequence where  $\overline{WR}$  occurs before  $\overline{ACK}$  and  $\overline{STB}$  occurs before  $\overline{RD}$  is permissible.  
 $(INTR = IBF \cdot \overline{MASK} \cdot \overline{STB} \cdot \overline{RD} + OBF \cdot \overline{MASK} \cdot \overline{ACK} \cdot \overline{WR})$



## 8251 BASIC FUNCTIONAL DESCRIPTION

### General

The 8251 is a Universal Synchronous/Asynchronous Receiver/Transmitter designed specifically for the 8080 Micro-computer System. Like other I/O devices in the 8080 Micro-computer System its functional configuration is programmed by the systems software for maximum flexibility. The 8251 can support virtually any serial data technique currently in use (including IBM "bi-sync").

In a communication environment an interface device must convert parallel format system data into serial format for transmission and convert incoming serial format data into parallel system data for reception. The interface device must also delete or insert bits or characters that are functionally unique to the communication technique. In essence, the interface should appear "transparent" to the CPU, a simple input or output of byte-oriented system data.

### Data Bus Buffer

This 3-state, bi-directional, 8-bit buffer is used to interface the 8251 to the 8080 system Data Bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions of the 8080 CPU. Control words, Command words and Status information are also transferred through the Data Bus Buffer.

### Read/Write Control Logic

This functional block accepts inputs from the 8080 Control bus and generates control signals for overall device operation. It contains the Control Word Register and Command Word Register that store the various control formats for device functional definition.

### RESET (Reset)

A "high" on this input forces the 8251 into an "Idle" mode. The device will remain at "Idle" until a new set of control words is written into the 8251 to program its functional definition. Minimum RESET pulse width is  $6 t_{CY}$ .

### CLK (Clock)

The CLK input is used to generate internal device timing and is normally connected to the Phase 2 (TTL) output of the 8224 Clock Generator. No external inputs or outputs are referenced to CLK but the frequency of CLK must be greater than 30 times the Receiver or Transmitter clock inputs for synchronous mode (4.5 times for asynchronous mode).

### $\overline{WR}$ (Write)

A "low" on this input informs the 8251 that the CPU is outputting data or control words, in essence, the CPU is writing out to the 8251.

### $\overline{RD}$ (Read)

A "low" on this input informs the 8251 that the CPU is inputting data or status information, in essence, the CPU is reading from the 8251.

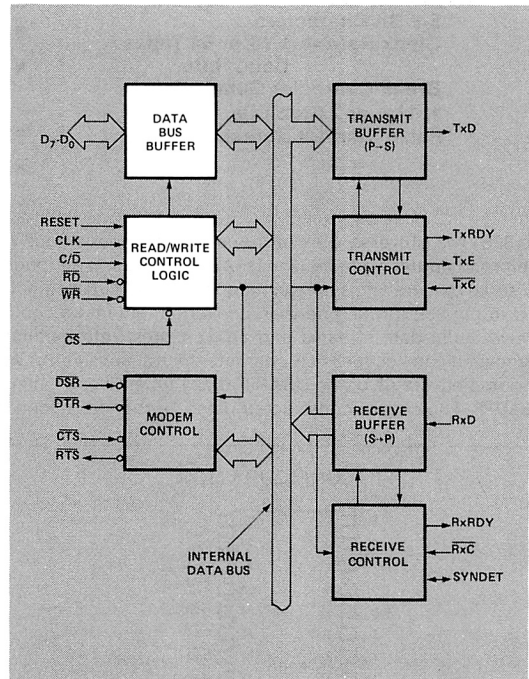
### $C/\overline{D}$ (Control/Data)

This input, in conjunction with the  $\overline{WR}$  and  $\overline{RD}$  inputs informs the 8251 that the word on the Data Bus is either a data character, control word or status information.

1 = CONTROL 0 = DATA

### $\overline{CS}$ (Chip Select)

A "low" on this input enables the 8251. No reading or writing will occur unless the device is selected.



$C/\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	0	1	0	8251 $\Rightarrow$ DATA BUS
0	1	0	0	DATA BUS $\Rightarrow$ 8251
1	0	1	0	STATUS $\Rightarrow$ DATA BUS
1	1	0	0	DATA BUS $\Rightarrow$ CONTROL
X	1	1	0	DATA BUS $\Rightarrow$ 3-STATE
X	X	X	1	DATA BUS $\Rightarrow$ 3-STATE

## Modem Control

The 8251 has a set of control inputs and outputs that can be used to simplify the interface to almost any Modem. The modem control signals are general purpose in nature and can be used for functions other than Modem control, if necessary.

### $\overline{\text{DSR}}$ (Data Set Ready)

The  $\overline{\text{DSR}}$  input signal is general purpose in nature. Its condition can be tested by the CPU using a Status Read operation. The  $\overline{\text{DSR}}$  input is normally used to test Modem conditions such as Data Set Ready.

### $\overline{\text{DTR}}$ (Data Terminal Ready)

The  $\overline{\text{DTR}}$  output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The  $\overline{\text{DTR}}$  output signal is normally used for Modem control such as Data Terminal Ready or Rate Select.

### $\overline{\text{RTS}}$ (Request to Send)

The  $\overline{\text{RTS}}$  output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The  $\overline{\text{RTS}}$  output signal is normally used for Modem control such as Request to Send.

### $\overline{\text{CTS}}$ (Clear to Send)

A "low" on this input enables the 8251 to transmit data (serial) if the Tx EN bit in the Command byte is set to a "one."

## Transmitter Buffer

The Transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the Tx<sub>D</sub> output pin.

## Transmitter Control

The Transmitter Control manages all activities associated with the transmission of serial data. It accepts and issues signals both externally and internally to accomplish this function.

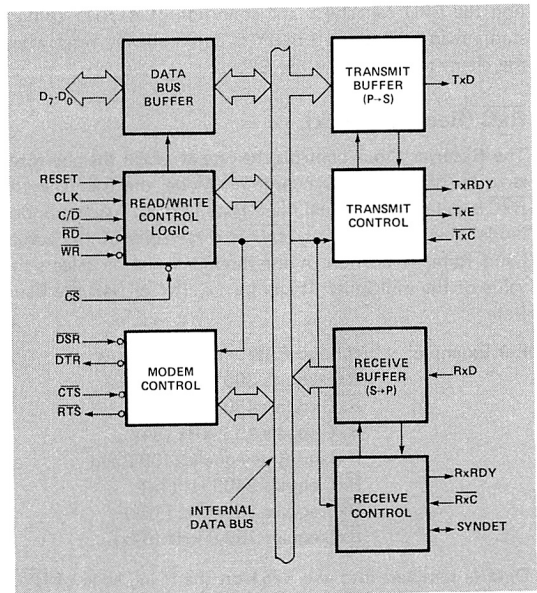
### TxDY (Transmitter Ready)

This output signals the CPU that the transmitter is ready to accept a data character. It can be used as an interrupt to the system or for the Polled operation the CPU can check TxDY using a status read operation. TxDY is automatically reset when a character is loaded from the CPU.

## TxE (Transmitter Empty)

When the 8251 has no characters to transmit, the TxE output will go "high". It resets automatically upon receiving a character from the CPU. TxE can be used to indicate the end of a transmission mode, so that the CPU "knows" when to "turn the line around" in the half-duplexed operational mode. TxE is independent of the TxEN bit in the Command instruction.

In SYNChronous mode, a "high" on this output indicates that a character has not been loaded and the SYNC character or characters are about to be transmitted automatically as "fillers". TxE goes low as soon as the SYNC is being shifted out.



## $\overline{\text{TxC}}$ (Transmitter Clock)

The Transmitter Clock controls the rate at which the character is to be transmitted. In the Synchronous transmission mode, the frequency of  $\overline{\text{TxC}}$  is equal to the actual Baud Rate (1X). In Asynchronous transmission mode, the frequency of  $\overline{\text{TxC}}$  is a multiple of the actual Baud Rate. A portion of the mode instruction selects the value of the multiplier; it can be 1x, 16x or 64x the Baud Rate.

For Example:

If Baud Rate equals 110 Baud,  
 $\overline{\text{TxC}}$  equals 110 Hz (1x)  
 $\overline{\text{TxC}}$  equals 1.76 kHz (16x)  
 $\overline{\text{TxC}}$  equals 7.04 kHz (64x).

The falling edge of  $\overline{\text{TxC}}$  shifts the serial data out of the 8251.

## Receiver Buffer

The Receiver accepts serial data, converts this serial input to parallel format, checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU. Serial data is input to the Rx $\overline{D}$  pin.

## Receiver Control

This functional block manages all receiver-related activities.

## RxRDY (Receiver Ready)

This output indicates that the 8251 contains a character that is ready to be input to the CPU. RxRDY can be connected to the interrupt structure of the CPU or for Polled operation the CPU can check the condition of RxRDY using a status read operation. RxRDY is automatically reset when the character is read by the CPU.

## RxC (Receiver Clock)

The Receiver Clock controls the rate at which the character is to be received. In Synchronous Mode, the frequency of  $\overline{RxC}$  is equal to the actual Baud Rate (1x). In Asynchronous Mode, the frequency of  $\overline{RxC}$  is a multiple of the actual Baud Rate. A portion of the mode instruction selects the value of the multiplier; it can be 1x, 16x or 64x the Baud Rate.

For Example: If Baud Rate equals 300 Baud,  
 $\overline{RxC}$  equals 300 Hz (1x)  
 $\overline{RxC}$  equals 4800 Hz (16x)  
 $\overline{RxC}$  equals 19.2 kHz (64x).  
 If Baud Rate equals 2400 Baud,  
 $\overline{RxC}$  equals 2400 Hz (1x)  
 $\overline{RxC}$  equals 38.4 kHz (16x)  
 $\overline{RxC}$  equals 153.6 kHz (64x).

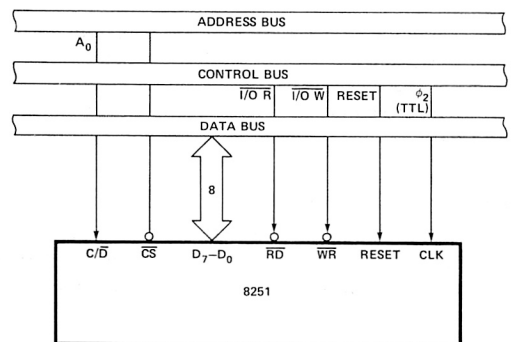
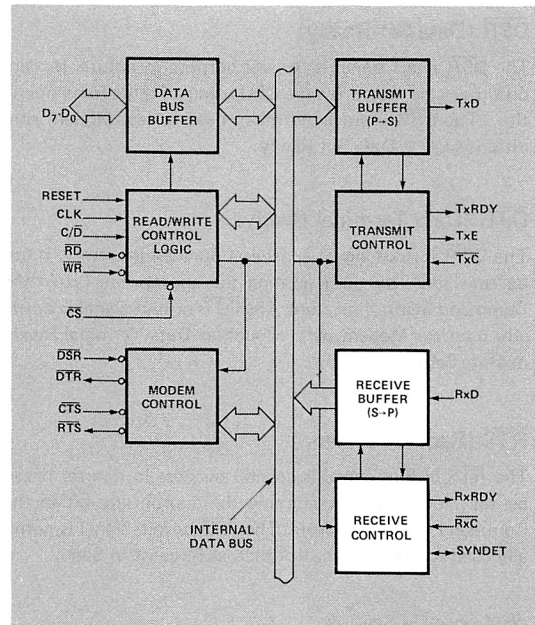
Data is sampled into the 8251 on the rising edge of  $\overline{RxC}$ .

NOTE: In most communications systems, the 8251 will be handling both the transmission and reception operations of a single link. Consequently, the Receive and Transmit Baud Rates will be the same. Both  $\overline{Tx\overline{C}}$  and  $\overline{RxC}$  will require identical frequencies for this operation and can be tied together and connected to a single frequency source (Baud Rate Generator) to simplify the interface.

## SYNDET (SYNC Detect)

This pin is used in SYNChronous Mode only. It is used as either input or output, programmable through the Control Word. It is reset to "low" upon RESET. When used as an output (internal Sync mode), the SYNDET pin will go "high" to indicate that the 8251 has located the SYNC character in the Receive mode. If the 8251 is programmed to use double Sync characters (bi-sync), then SYNDET will go "high" in the middle of the last bit of the second Sync character. SYNDET is automatically reset upon a Status Read operation.

When used as an input, (external SYNC detect mode), a positive going signal will cause the 8251 to start assembling data characters on the falling edge of the next  $\overline{RxC}$ . Once in SYNC, the "high" input signal can be removed. The duration of the high signal should be at least equal to the period of  $\overline{RxC}$ .



8251 Interface to 8080 Standard System Bus



## DETAILED OPERATION DESCRIPTION

### General

The complete functional definition of the 8251 is programmed by the systems software. A set of control words must be sent out by the CPU to initialize the 8251 to support the desired communications format. These control words will program the: BAUD RATE, CHARACTER LENGTH, NUMBER OF STOP BITS, SYNCHRONOUS or ASYNCHRONOUS OPERATION, EVEN/ODD PARITY etc. In the Synchronous Mode, options are also provided to select either internal or external character synchronization.

Once programmed, the 8251 is ready to perform its communication functions. The TxRDY output is raised "high" to signal the CPU that the 8251 is ready to receive a character. This output (TxRDY) is reset automatically when the CPU writes a character into the 8251. On the other hand, the 8251 receives serial data from the MODEM or I/O device, upon receiving an entire character the RxRDY output is raised "high" to signal the CPU that the 8251 has a complete character ready for the CPU to fetch. RxRDY is reset automatically upon the CPU read operation.

The 8251 cannot begin transmission until the TxEN (Transmitter Enable) bit is set in the Command Instruction and it has received a Clear To Send (CTS) input. The Tx output will be held in the marking state upon Reset.

### Programming the 8251

Prior to starting data transmission or reception, the 8251 must be loaded with a set of control words generated by the CPU. These control signals define the complete functional definition of the 8251 and must immediately follow a Reset operation (internal or external).

The control words are split into two formats:

1. Mode Instruction
2. Command Instruction

#### Mode Instruction

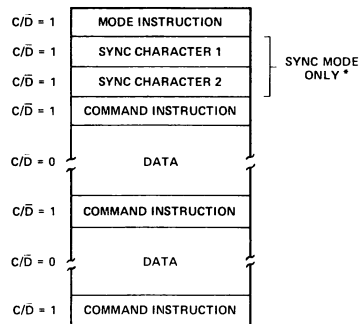
This format defines the general operational characteristics of the 8251. It must follow a Reset operation (internal or external). Once the Mode instruction has been written into the 8251 by the CPU, SYNC characters or Command instructions may be inserted.

#### Command Instruction

This format defines a status word that is used to control the actual operation of the 8251.

Both the Mode and Command instructions must conform to a specified sequence for proper device operation. The Mode Instruction must be inserted immediately following a Reset operation, prior to using the 8251 for data communication.

All control words written into the 8251 after the Mode Instruction will load the Command Instruction. Command Instructions can be written into the 8251 at any time in the data block during the operation of the 8251. To return to the Mode Instruction format a bit in the Command Instruction word can be set to initiate an internal Reset operation which automatically places the 8251 back into the Mode Instruction format. Command Instructions must follow the Mode Instructions or Sync characters.



\*The second SYNC character is skipped if MODE instruction has programmed the 8251 to single character Internal SYNC Mode. Both SYNC characters are skipped if MODE instruction has programmed the 8251 to ASYNC mode.

### Typical Data Block

## Mode Instruction Definition

The 8251 can be used for either Asynchronous or Synchronous data communication. To understand how the Mode Instruction defines the functional operation of the 8251 the designer can best view the device as two separate components sharing the same package. One Asynchronous the other Synchronous. The format definition can be changed "on the fly" but for explanation purposes the two formats will be isolated.

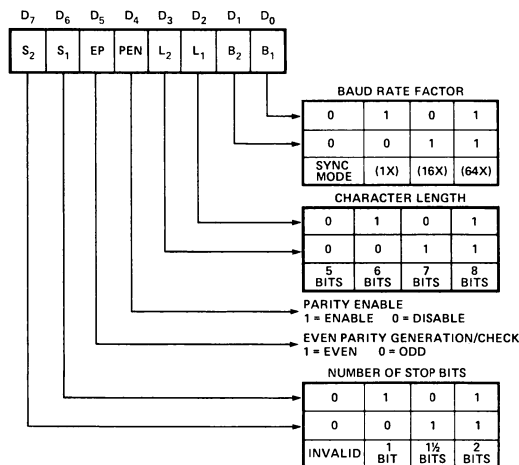
### Asynchronous Mode (Transmission)

Whenever a data character is sent by the CPU the 8251 automatically adds a Start bit (low level) and the programmed number of Stop bits to each character. Also, an even or odd Parity bit is inserted prior to the Stop bit(s), as defined by the Mode Instruction. The character is then transmitted as a serial data stream on the TxD output. The serial data is shifted out on the falling edge of  $\overline{\text{TxC}}$  at a rate equal to 1, 1/16, or 1/64 that of the  $\overline{\text{TxC}}$ , as defined by the Mode Instruction. BREAK characters can be continuously sent to the TxD if commanded to do so.

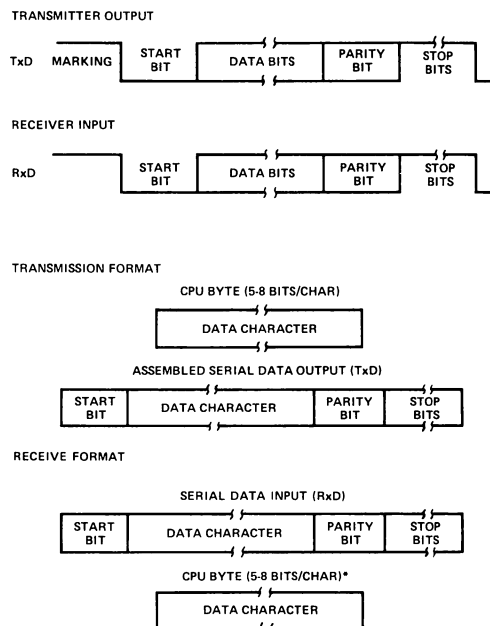
When no data characters have loaded into the 8251 the TxD output remains "high" (marking) unless a Break (continuously low) has been programmed.

### Asynchronous Mode (Receive)

The RxD line is normally high. A falling edge on this line triggers the beginning of a START bit. The validity of this START bit is checked by again strobing this bit at its nominal center. If a low is detected again, it is a valid START bit, and the bit counter will start counting. The bit counter locates the center of the data bits, the parity bit (if it exists) and the stop bits. If parity error occurs, the parity error flag is set. Data and parity bits are sampled on the RxD pin with the rising edge of  $\overline{\text{RxC}}$ . If a low level is detected as the STOP bit, the Framing Error flag will be set. The STOP bit signals the end of a character. This character is then loaded into the parallel I/O buffer of the 8251. The RxRDY pin is raised to signal the CPU that a character is ready to be fetched. If a previous character has not been fetched by the CPU, the present character replaces it in the I/O buffer, and the OVERRUN flag is raised (thus the previous character is lost). All of the error flags can be reset by a command instruction. The occurrence of any of these errors will not stop the operation of the 8251.



### Mode Instruction Format, Asynchronous Mode



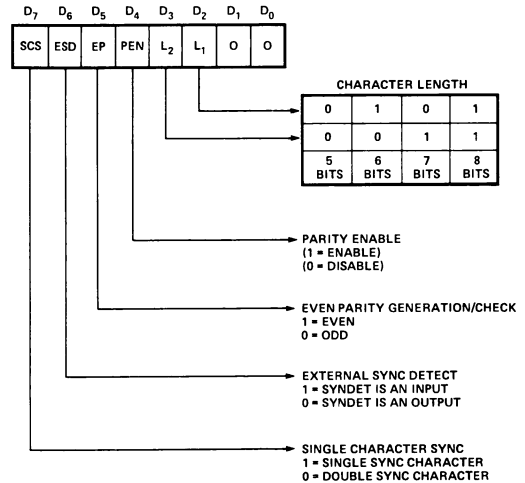
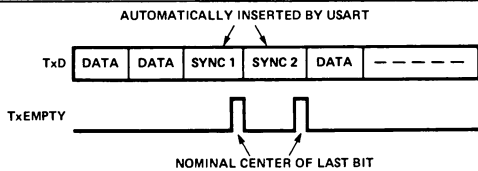
\*NOTE: IF CHARACTER LENGTH IS DEFINED AS 5, 6 OR 7 BITS THE UNUSED BITS ARE SET TO "ZERO".

### Asynchronous Mode

## Synchronous Mode (Transmission)

The TxD output is continuously high until the CPU sends its first character to the 8251 which usually is a SYNC character. When the  $\overline{\text{CTS}}$  line goes low, the first character is serially transmitted out. All characters are shifted out on the falling edge of  $\overline{\text{TxC}}$ . Data is shifted out at the same rate as the  $\overline{\text{TxC}}$ .

Once transmission has started, the data stream at TxD output must continue at the  $\overline{\text{TxC}}$  rate. If the CPU does not provide the 8251 with a character before the 8251 becomes empty, the SYNC characters (or character if in single SYNC word mode) will be automatically inserted in the TxD data stream. In this case, the TxEMPTY pin is raised high to signal that the 8251 is empty and SYNC characters are being sent out. TxEMPTY goes low when SYNC is being shifted out (See Figure below). The TxEMPTY pin is internally reset by the next character being written into the 8251.



## Mode Instruction Format, Synchronous Mode

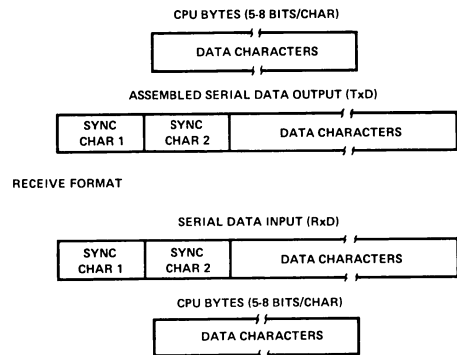
## Synchronous Mode (Receive)

In this mode, character synchronization can be internally or externally achieved. If the internal SYNC mode has been programmed, the receiver starts in a HUNT mode. Data on the RxD pin is then sampled in on the rising edge of  $\overline{\text{RxC}}$ . The content of the Rx buffer is continuously compared with the first SYNC character until a match occurs. If the 8251 has been programmed for two SYNC characters, the subsequent received character is also compared; when both SYNC characters have been detected, the USART ends the HUNT mode and is in character synchronization. The SYNDET pin is then set high, and is reset automatically by a STATUS READ.

In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin. The high level can be removed after one  $\overline{\text{RxC}}$  cycle.

Parity error and overrun error are both checked in the same way as in the Asynchronous Rx mode.

The CPU can command the receiver to enter the HUNT mode if synchronization is lost.

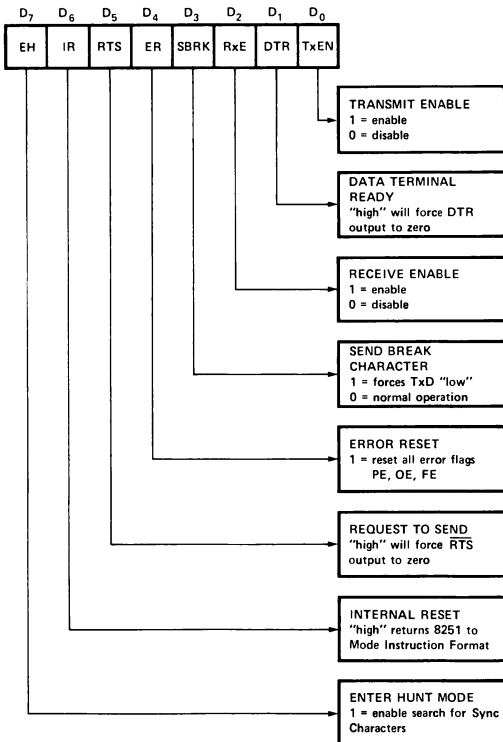


## Synchronous Mode, Transmission Format

## COMMAND INSTRUCTION DEFINITION

Once the functional definition of the 8251 has been programmed by the Mode Instruction and the Sync Characters are loaded (if in Sync Mode) then the device is ready to be used for data communication. The Command Instruction controls the actual operation of the selected format. Functions such as: Enable Transmit/Receive, Error Reset and Modem Controls are provided by the Command Instruction.

Once the Mode Instruction has been written into the 8251 and Sync characters inserted, if necessary, then all further "control writes" ( $C/\overline{D} = 1$ ) will load the Command Instruction. A Reset operation (internal or external) will return the 8251 to the Mode Instruction Format.



Command Instruction Format

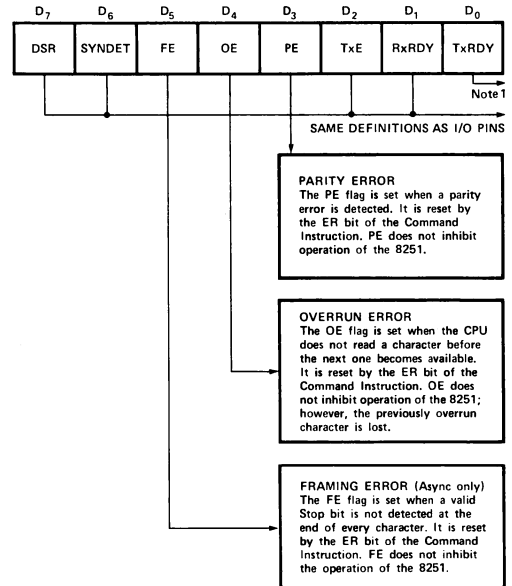
## STATUS READ DEFINITION

In data communication systems it is often necessary to examine the "status" of the active device to ascertain if errors have occurred or other conditions that require the processor's attention. The 8251 has facilities that allow the programmer to "read" the status of the device at any time during the functional operation.

A normal "read" command is issued by the CPU with the C/D input at one to accomplish this function.

Some of the bits in the Status Read Format have identical meanings to external output pins so that the 8251 can be used in a completely Polled environment or in an interrupt driven environment.

Status update can have a maximum delay of 16 clock periods.



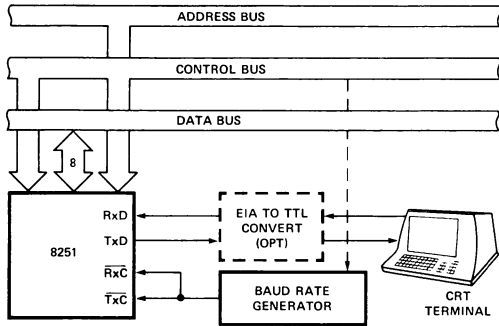
### Status Read Format

Note 1: TxRDY status bit has similar meaning as the TxRDY output pin. The former is not conditioned by CTS and TxEN; the latter is conditioned by both CTS and TxEN.

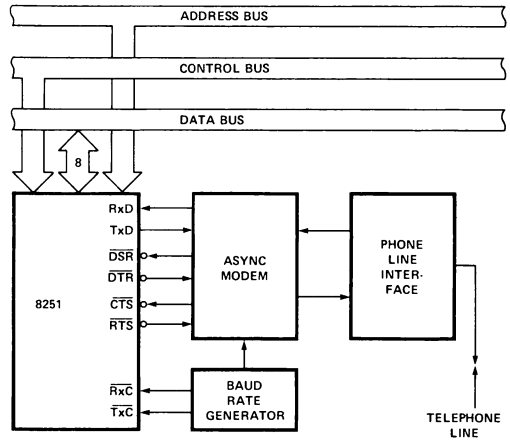
i.e. TxRDY status bit = DB Buffer Empty

TxRDY pin out = DB Buffer Empty • CTS • TxEN

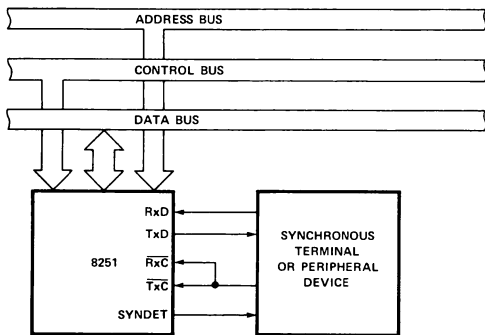
## APPLICATIONS OF THE 8251



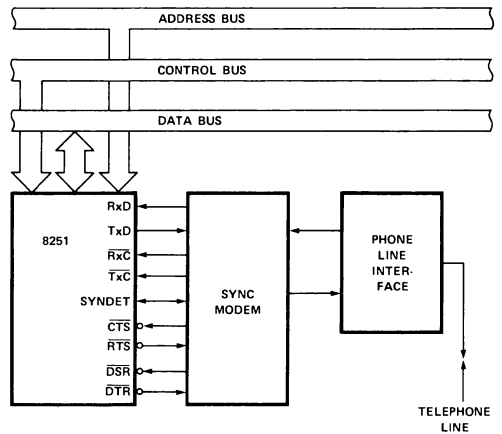
Asynchronous Serial Interface to CRT Terminal,  
DC-9600 Baud



Asynchronous Interface to Telephone Lines



Synchronous Interface to Terminal or Peripheral Device



Synchronous Interface to Telephone Lines

## Absolute Maximum Ratings\*

Ambient Temperature Under Bias. . . . . 0°C to 70°C  
 Storage Temperature . . . . . -65°C to +150°C  
 Voltage On Any Pin  
     With Respect to Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

*\*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

## D.C. Characteristics:

$T_A = 0^\circ\text{C to } 70^\circ\text{C}$ ;  $V_{CC} = 5.0\text{V} \pm 5\%$ ;  $\text{GND} = 0\text{V}$

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
$V_{IL}$	Input Low Voltage	-.5		0.8	V	
$V_{IH}$	Input High Voltage	2.0		$V_{CC}$	V	
$V_{OL}$	Output Low Voltage			0.45	V	$I_{OL} = 1.6\text{mA}$
$V_{OH}$	Output High Voltage	2.4			V	$I_{OH} = -100\mu\text{A}$
$I_{DL}$	Data Bus Leakage			-50 10	$\mu\text{A}$ $\mu\text{A}$	$V_{OUT} = .45\text{V}$ $V_{OUT} = V_{CC}$
$I_{IL}$	Input Leakage			10	$\mu\text{A}$	$V_{IN} = V_{CC}$
$I_{CC}$	Power Supply Current		45	80	mA	

## Capacitance:

$T_A = 25^\circ\text{C}$ ;  $V_{CC} = \text{GND} = 0\text{V}$

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
$C_N$	Input Capacitance			10	pF	$f_c = 1\text{MHz}$
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to GND.

## TEST LOAD CIRCUIT:

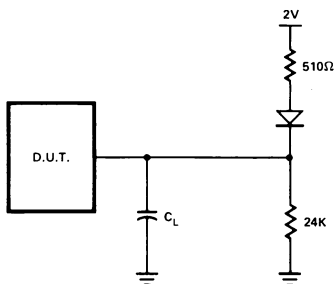
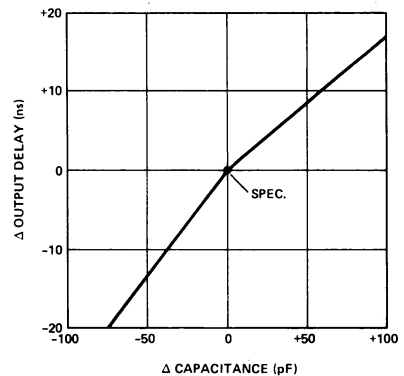


Figure 1.

## TYPICAL $\Delta$ OUTPUT DELAY VS. $\Delta$ CAPACITANCE (dB)



## A.C. Characteristics:

$T_A = 0^{\circ}\text{C}$  to  $70^{\circ}\text{C}$ ;  $V_{CC} = 5.0\text{V} \pm 5\%$ ;  $\text{GND} = 0\text{V}$

### BUS PARAMETERS: (Note 1)

#### READ CYCLE

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AR}$	Address Stable Before $\overline{\text{READ}}$ ( $\overline{\text{CS}}$ , C/D)	50		ns	
$t_{RA}$	Address Hold Time for $\overline{\text{READ}}$ ( $\overline{\text{CS}}$ , C/D)	5		ns	
$t_{RR}$	$\overline{\text{READ}}$ Pulse Width	430		ns	
$t_{RD}$	Data Delay from $\overline{\text{READ}}$		350	ns	$C_L = 100\text{ pF}$
$t_{DF}$	$\overline{\text{READ}}$ to Data Floating		200	ns	$C_L = 100\text{ pF}$
		25		ns	$C_L = 15\text{ pF}$
$t_{RV}$	Recovery Time Between WRITES (Note 2)	6		$t_{CY}$	

#### WRITE CYCLE

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AW}$	Address Stable Before $\overline{\text{WRITE}}$	20		ns	
$t_{WA}$	Address Hold Time for $\overline{\text{WRITE}}$	20		ns	
$t_{WW}$	$\overline{\text{WRITE}}$ Pulse Width	400		ns	
$t_{DW}$	Data Set Up Time for $\overline{\text{WRITE}}$	200		ns	
$t_{WD}$	Data Hold Time for $\overline{\text{WRITE}}$	40		ns	

NOTES: 1. AC timings measured at  $V_{OH} = 2.0$ ,  $V_{OL} = .8$ , and with load circuit of Figure 1.  
 2. This recovery time is for initialization only, when MODE, SYNC1, SYNC2, COMMAND and first DATA BYTES are written into the USART. Subsequent writing of both COMMAND and DATA are only allowed when  $\text{TxRDY} = 1$ .

## OTHER TIMINGS:

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{CY}$	Clock Period (Note 3)	.420	1.35	$\mu s$	
$t_{\phi W}$	Clock Pulse Width	220	.7 $t_{CY}$	ns	
$t_R, t_F$	Clock Rise and Fall Time	0	50	ns	
$t_{DTx}$	TxD Delay from Falling Edge of Tx C		1	$\mu s$	$C_L = 100 \text{ pF}$
$t_{SRx}$	Rx Data Set-Up Time to Sampling Pulse	2		$\mu s$	$C_L = 100 \text{ pF}$
$t_{HRx}$	Rx Data Hold Time to Sampling Pulse	2		$\mu s$	$C_L = 100 \text{ pF}$
$f_{Tx}$	Transmitter Input Clock Frequency				
	1x Baud Rate	DC	56	KHz	
	16x and 64x Baud Rate	DC	520	KHz	
$t_{TPW}$	Transmitter Input Clock Pulse Width				
	1x Baud Rate	12		$t_{CY}$	
	16x and 64x Baud Rate	1		$t_{CY}$	
$t_{TPD}$	Transmitter Input Clock Pulse Delay				
	1x Baud Rate	15		$t_{CY}$	
	16x and 64x Baud Rate	3		$t_{CY}$	
$f_{Rx}$	Receiver Input Clock Frequency				
	1x Baud Rate	DC	56	KHz	
	16x and 64x Baud Rate	DC	520	KHz	
$t_{RPW}$	Receiver Input Clock Pulse Width				
	1x Baud Rate	12		$t_{CY}$	
	16x and 64x Baud Rate	1		$t_{CY}$	
$t_{RPD}$	Receiver Input Clock Pulse Delay				
	1x Baud Rate	15		$t_{CY}$	
	16x and 64x Baud Rate	3		$t_{CY}$	
$t_{Tx}$	TxRDY Delay from Center of Data Bit		16	$t_{CY}$	$C_L = 50 \text{ pF}$
$t_{Rx}$	RxRDY Delay from Center of Data Bit		20	$t_{CY}$	
$t_{IS}$	Internal SYNDET Delay from Center of Data Bit		25	$t_{CY}$	
$t_{ES}$	Internal SYNDET Set-Up Time Before Falling Edge of Rx C		16	$t_{CY}$	
$t_{Tx E}$	TxEMPTY Delay from Center of Data Bit		16	$t_{CY}$	$C_L = 50 \text{ pF}$
$t_{WC}$	Control Delay from Rising Edge of WRITE (Tx E, DTR, RTS)		16	$t_{CY}$	
$t_{CR}$	Control to READ Set-Up Time ( $\overline{DSR}$ , $\overline{CTS}$ )		16	$t_{CY}$	

3. The Tx C and Rx C frequencies have the following limitations with respect to CLK.

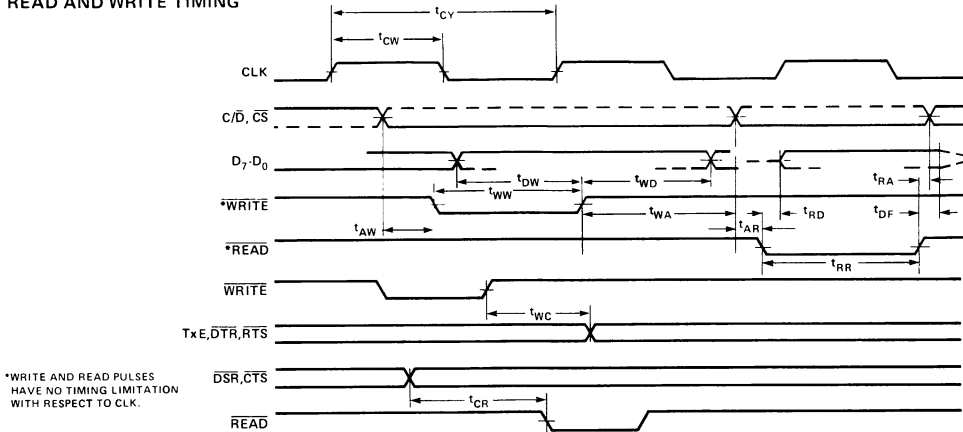
For 1x Baud Rate,  $f_{Tx}$  or  $f_{Rx} \leq 1/(30 t_{CY})$

For 16x and 64x Baud Rate,  $f_{Tx}$  or  $f_{Rx} \leq 1/(4.5 t_{CY})$

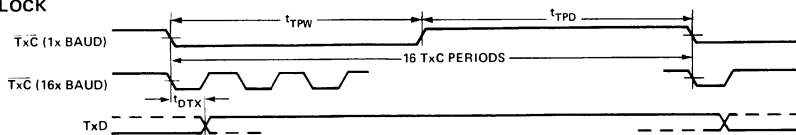
4. Reset Pulse Width = 6  $t_{CY}$  minimum.



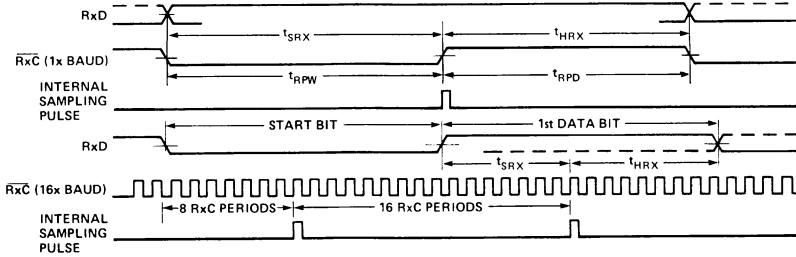
## READ AND WRITE TIMING



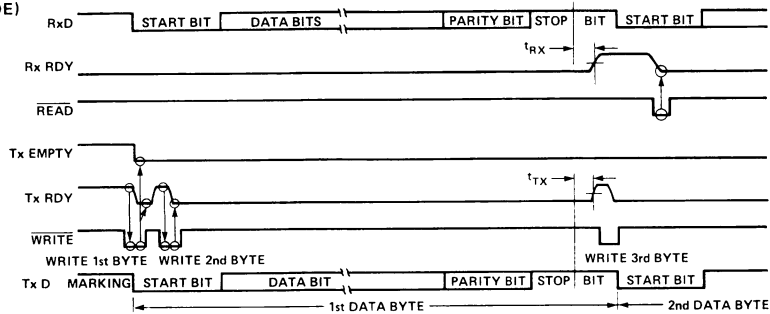
## TRANSMITTER CLOCK AND DATA



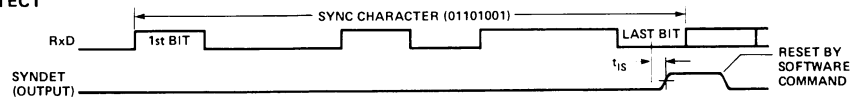
## RECEIVER CLOCK AND DATA



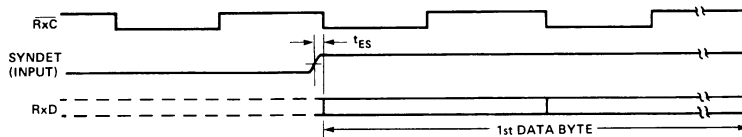
## Tx RDY AND Rx RDY TIMING (ASYNC MODE)



## INTERNAL SYNC DETECT



## EXTERNAL SYNC DETECT





## 8205

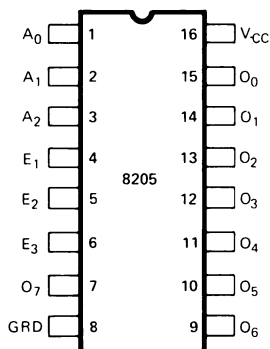
### HIGH SPEED 1 OUT OF 8 BINARY DECODER

- I/O Port or Memory Selector
- Simple Expansion — Enable Inputs
- High Speed Schottky Bipolar Technology — 18ns Max. Delay
- Directly Compatible with TTL Logic Circuits
- Low Input Load Current — .25 mA max., 1/6 Standard TTL Input Load
- Minimum Line Reflection — Low Voltage Diode Input Clamp
- Outputs Sink 10 mA min.
- 16-Pin Dual-In-Line Ceramic or Plastic Package

The 8205 decoder can be used for expansion of systems which utilize input ports, output ports, and memory components with active low chip select input. When the 8205 is enabled, one of its eight outputs goes "low", thus a single row of a memory system is selected. The 3 chip enable inputs on the 8205 allow easy system expansion. For very large systems, 8205 decoders can be cascaded such that each decoder can drive eight other decoders for arbitrary memory expansions.

The Intel® 8205 is packaged in a standard 16 pin dual-in-line package; and its performance is specified over the temperature range of 0°C to +75°C, ambient. The use of Schottky barrier diode clamped transistors to obtain fast switching speeds results in higher performance than equivalent devices made with a gold diffusion process.

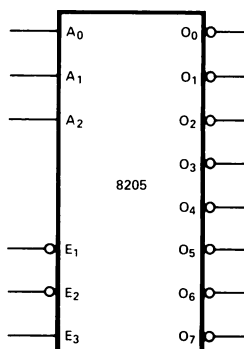
PIN CONFIGURATION



PIN NAMES

A <sub>0</sub> A <sub>2</sub>	ADDRESS INPUTS
E <sub>1</sub> E <sub>3</sub>	ENABLE INPUTS
O <sub>0</sub> O <sub>7</sub>	DECODED OUTPUTS

LOGIC SYMBOL



ADDRESS			ENABLE			OUTPUTS							
A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	E <sub>1</sub>	E <sub>2</sub>	E <sub>3</sub>	0	1	2	3	4	5	6	7
L	L	L	L	L	H	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
L	H	L	L	L	H	H	H	L	H	H	H	H	H
H	H	L	L	L	H	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
H	L	H	L	L	H	H	H	H	H	H	L	H	H
L	H	H	L	L	H	H	H	H	H	H	H	L	H
H	H	H	L	L	H	H	H	H	H	H	H	H	L
X	X	X	L	L	L	H	H	H	H	H	H	H	H
X	X	X	H	L	L	H	H	H	H	H	H	H	H
X	X	X	L	H	L	H	H	H	H	H	H	H	H
X	X	X	H	H	L	H	H	H	H	H	H	H	H
X	X	X	H	L	H	H	H	H	H	H	H	H	H
X	X	X	L	H	H	H	H	H	H	H	H	H	H
X	X	X	H	H	H	H	H	H	H	H	H	H	H

## FUNCTIONAL DESCRIPTION

### Decoder

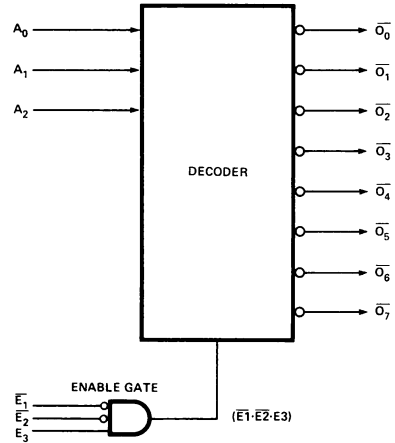
The 8205 contains a one out of eight binary decoder. It accepts a three bit binary code and by gating this input, creates an exclusive output that represents the value of the input code.

For example, if a binary code of 101 was present on the A0, A1 and A2 address input lines, and the device was enabled, an active low signal would appear on the  $\overline{O_5}$  output line. Note that all of the other output pins are sitting at a logic high, thus the decoded output is said to be exclusive. The decoders outputs will follow the truth table shown below in the same manner for all other input variations.

### Enable Gate

When using a decoder it is often necessary to gate the outputs with timing or enabling signals so that the exclusive output of the decoded value is synchronous with the overall system.

The 8205 has a built-in function for such gating. The three enable inputs ( $\overline{E_1}$ ,  $\overline{E_2}$ , E3) are ANDed together and create a single enable signal for the decoder. The combination of both active "high" and active "low" device enable inputs provides the designer with a powerfully flexible gating function to help reduce package count in his system.



ADDRESS			ENABLE			OUTPUTS							
A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	E <sub>1</sub>	E <sub>2</sub>	E <sub>3</sub>	0	1	2	3	4	5	6	7
L	L	L	L	L	H	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
L	H	L	L	L	H	H	H	L	H	H	H	H	H
H	H	L	L	L	H	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
H	L	H	L	L	H	H	H	H	H	H	L	H	H
L	H	H	L	L	H	H	H	H	H	H	H	L	H
H	H	H	L	L	H	H	H	H	H	H	H	H	L
X	X	X	L	L	L	H	H	H	H	H	H	H	H
X	X	X	H	L	L	H	H	H	H	H	H	H	H
X	X	X	L	H	L	H	H	H	H	H	H	H	H
X	X	X	H	H	L	H	H	H	H	H	H	H	H
X	X	X	H	L	H	H	H	H	H	H	H	H	H
X	X	X	L	H	H	H	H	H	H	H	H	H	H
X	X	X	H	H	H	H	H	H	H	H	H	H	H

## APPLICATIONS OF THE 8205

The 8205 can be used in a wide variety of applications in microcomputer systems. I/O ports can be decoded from the address bus, chip select signals can be generated to select memory devices and the type of machine state such as in 8008 systems can be derived from a simple decoding of the state lines (S0, S1, S2) of the 8008 CPU.

### I/O Port Decoder

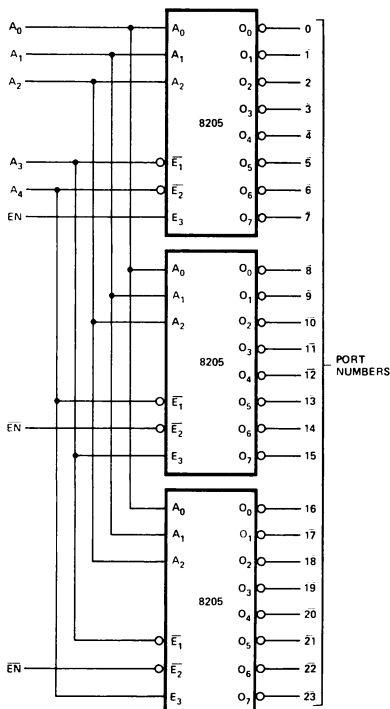
Shown in the figure below is a typical application of the 8205. Address input lines are decoded by a group of 8205s (3). Each input has a binary weight. For example, A0 is assigned a value of 1 and is the LSB; A4 is assigned a value of 16 and is the MSB. By connecting them to the decoders as shown, an active low signal that is exclusive in nature and represents the value of the input address lines, is available at the outputs of the 8205s.

This circuit can be used to generate enable signals for I/O ports or any other decoder related application.

Note that no external gating is required to decode up to 24 exclusive devices and that a simple addition of an inverter or two will allow expansion to even larger decoder networks.

### Chip Select Decoder

Using a very similar circuit to the I/O port decoder, an ar-



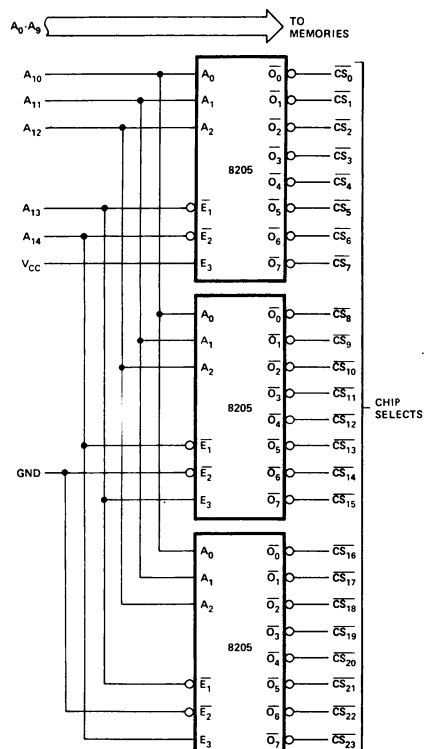
I/O Port Decoder

ray of 8205s can be used to create a simple interface to a 24K memory system.

The memory devices used can be either ROM or RAM and are 1K in storage capacity. 8308s and 8102s are the devices typically used for this application. This type of memory device has ten (10) address inputs and an active "low" chip select ( $\overline{CS}$ ). The lower order address bits A0-A9 which come from the microprocessor are "bussed" to all memory elements and the chip select to enable a specific device or group of devices comes from the array of 8205s. The output of the 8205 is active low so it is directly compatible with the memory components.

Basic operation is that the CPU issues an address to identify a specific memory location in which it wishes to "write" or "read" data. The most significant address bits A10-A14 are decoded by the array of 8205s and an exclusive, active low, chip select is generated that enables a specific memory device. The least significant address bits A0-A9 identify a specific location within the selected device. Thus, all addresses throughout the entire memory array are exclusive in nature and are non-redundant.

This technique can be expanded almost indefinitely to support even larger systems with the addition of a few inverters and an extra decoder (8205).



24K Memory Interface

## Logic Element Example

Probably the most overlooked application of the 8205 is that of a general purpose logic element. Using the "on-chip" enabling gate, the 8205 can be configured to gate its decoded outputs with system timing signals and generate strobes that can be directly connected to latches, flip-flops and one-shots that are used throughout the system.

An excellent example of such an application is the "state decoder" in an 8008 CPU based system. The 8008 CPU issues three bits of information (S<sub>0</sub>, S<sub>1</sub>, S<sub>2</sub>) that indicate the nature of the data on the Data Bus during each machine state. Decoding of these signals is vital to generate strobes that can load the address latches, control bus discipline and general machine functions.

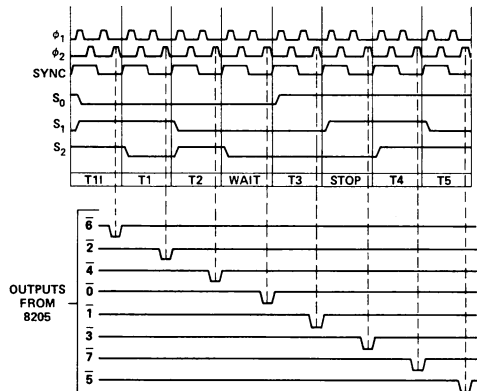
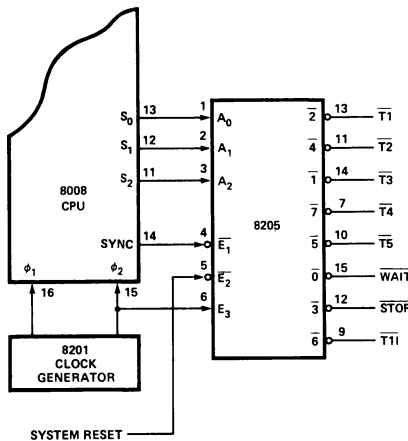
In the figure below a circuit is shown using the 8205 as the "state decoder" for an 8008 CPU that not only decodes the S<sub>0</sub>, S<sub>1</sub>, S<sub>2</sub> outputs but gates these signals with the clock (phase 2) and the SYNC output of the 8008 CPU. The  $\overline{T1}$

and  $\overline{T2}$  decoded strobes can connect directly to devices like 8212s for latching the address information. The other decoded strobes can be used to generate signals to control the system data bus, memory timing functions and interrupt structure. RESET is connected to the enable gate so that strobes are not generated during system reset, eliminating accidental loading.

The power of such a circuit becomes evident when a single decoded strobe is logically broken down. Consider  $\overline{T1}$  output, the boolean equation for it would be:

$$\overline{T1} = (\overline{S_0} \cdot S_1 \cdot \overline{S_2}) \cdot (\overline{SYNC} \cdot \text{Phase 2} \cdot \overline{\text{Reset}})$$

A six input NAND gate plus a few inverters would be needed to implement this function. The seven remaining outputs would need a similar circuit to duplicate their function, obviously a substantial savings in components can be achieved when using such a technique.



State Control Coding

S <sub>0</sub>	S <sub>1</sub>	S <sub>2</sub>	STATE
0	1	0	T <sub>1</sub>
0	1	1	T <sub>11</sub>
0	0	1	T <sub>2</sub>
0	0	0	WAIT
1	0	0	T <sub>3</sub>
1	1	0	STOP
1	1	1	T <sub>4</sub>
1	0	1	T <sub>5</sub>

**ABSOLUTE MAXIMUM RATINGS\***

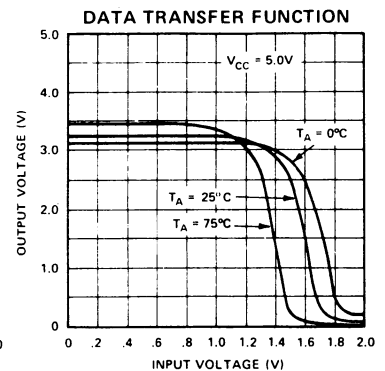
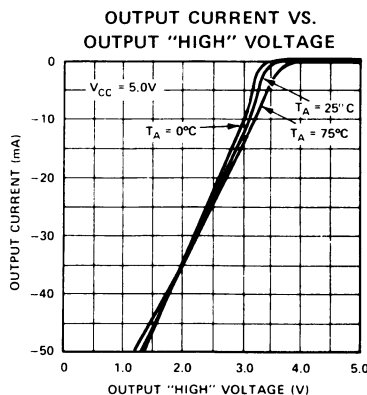
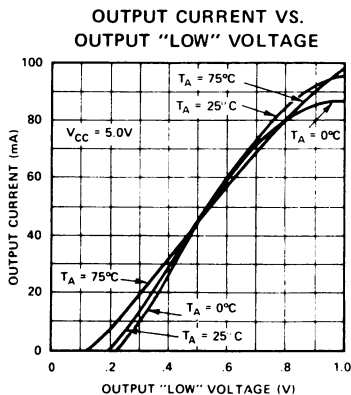
Temperature Under Bias:	Ceramic	-65°C to +125°C
	Plastic	-65°C to +75°C
Storage Temperature		-65°C to +160°C
All Output or Supply Voltages		-0.5 to +7 Volts
All Input Voltages		-1.0 to +5.5 Volts
Output Currents		125 mA

**\*COMMENT**

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

**D.C. CHARACTERISTICS**  $T_A = 0^\circ\text{C}$  to  $+75^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V} \pm 5\%$ **8205**

SYMBOL	PARAMETER	LIMIT		UNIT	TEST CONDITIONS
		MIN.	MAX.		
$I_F$	INPUT LOAD CURRENT		-0.25	mA	$V_{CC} = 5.25\text{V}$ , $V_F = 0.45\text{V}$
$I_R$	INPUT LEAKAGE CURRENT		10	$\mu\text{A}$	$V_{CC} = 5.25\text{V}$ , $V_R = 5.25\text{V}$
$V_C$	INPUT FORWARD CLAMP VOLTAGE		-1.0	V	$V_{CC} = 4.75\text{V}$ , $I_C = -5.0\text{ mA}$
$V_{OL}$	OUTPUT "LOW" VOLTAGE		0.45	V	$V_{CC} = 4.75\text{V}$ , $I_{OL} = 10.0\text{ mA}$
$V_{OH}$	OUTPUT HIGH VOLTAGE	2.4		V	$V_{CC} = 4.75\text{V}$ , $I_{OH} = -1.5\text{ mA}$
$V_{IL}$	INPUT "LOW" VOLTAGE		0.85	V	$V_{CC} = 5.0\text{V}$
$V_{IH}$	INPUT "HIGH" VOLTAGE	2.0		V	$V_{CC} = 5.0\text{V}$
$I_{SC}$	OUTPUT HIGH SHORT CIRCUIT CURRENT	-40	-120	mA	$V_{CC} = 5.0\text{V}$ , $V_{OUT} = 0\text{V}$
$V_{OX}$	OUTPUT "LOW" VOLTAGE @ HIGH CURRENT		0.8	V	$V_{CC} = 5.0\text{V}$ , $I_{OX} = 40\text{ mA}$
$I_{CC}$	POWER SUPPLY CURRENT		70	mA	$V_{CC} = 5.25\text{V}$

**TYPICAL CHARACTERISTICS**

## 8205 SWITCHING CHARACTERISTICS

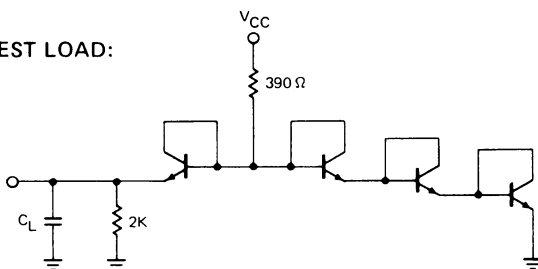
## CONDITIONS OF TEST:

Input pulse amplitudes: 2.5V

Input rise and fall times: 5 nsec  
between 1V and 2V

Measurements are made at 1.5V

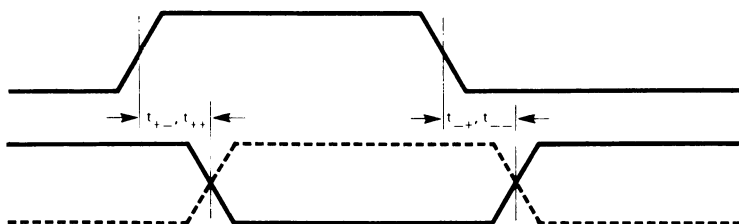
## TEST LOAD:

All Transistors 2N2369 or Equivalent.  $C_L = 30 \text{ pF}$ 

## TEST WAVEFORMS

ADDRESS OR ENABLE  
INPUT PULSE

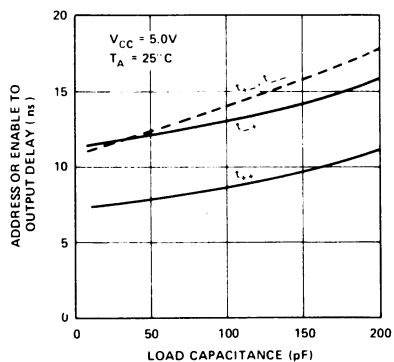
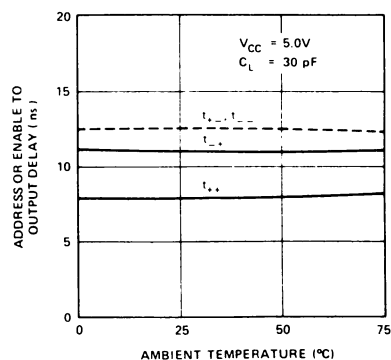
OUTPUT

A.C. CHARACTERISTICS  $T_A = 0^\circ\text{C}$  to  $+75^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V} \pm 5\%$  unless otherwise specified.

SYMBOL	PARAMETER	MAX. LIMIT	UNIT	TEST CONDITIONS
$t_{++}$	ADDRESS OR ENABLE TO OUTPUT DELAY	18	ns	
$t_{-+}$		18	ns	
$t_{+-}$		18	ns	
$t_{--}$		18	ns	
$C_{IN}^{(1)}$	INPUT CAPACITANCE	P8205 5(typ.)	pF	$f = 1 \text{ MHz}$ , $V_{CC} = 0\text{V}$ $V_{BIAS} = 2.0\text{V}$ , $T_A = 25^\circ\text{C}$
		C8205 5(typ.)	pF	

1. This parameter is periodically sampled and is not 100% tested.

## TYPICAL CHARACTERISTICS

ADDRESS OR ENABLE TO OUTPUT  
DELAY VS. LOAD CAPACITANCEADDRESS OR ENABLE TO OUTPUT  
DELAY VS. AMBIENT TEMPERATURE



## PRIORITY INTERRUPT CONTROL UNIT

- Eight Priority Levels
- Current Status Register
- Priority Comparator
- Fully Expandable
- High Performance (50ns)
- 24-Pin Dual In-Line Package

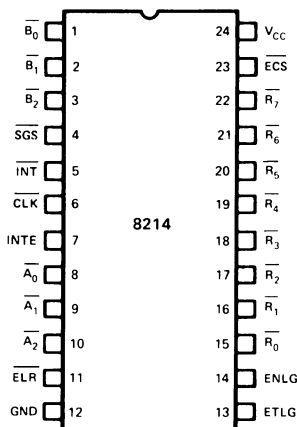
The 8214 is an eight level priority interrupt control unit designed to simplify interrupt driven microcomputer systems.

The PICU can accept eight requesting levels; determine the highest priority, compare this priority to a software controlled current status register and issue an interrupt to the system along with vector information to identify the service routine.

The 8214 is fully expandable by the use of open collector interrupt output and vector information. Control signals are also provided to simplify this function.

The PICU is designed to support a wide variety of vectored interrupt structures and reduce package count in interrupt driven microcomputer systems.

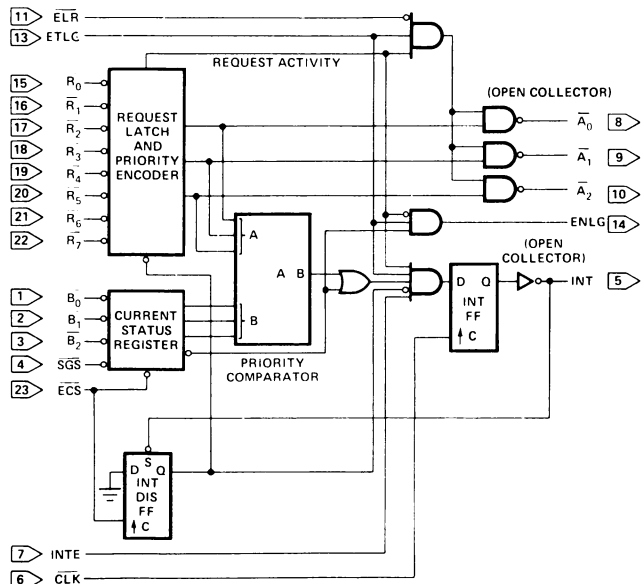
### PIN CONFIGURATION



### PIN NAMES

INPUTS	
$\overline{R_0}-\overline{R_7}$	REQUEST LEVELS ( $R_7$ HIGHEST PRIORITY)
$\overline{B_0}-\overline{B_2}$	CURRENT STATUS
SGS	STATUS GROUP SELECT
ECS	ENABLE CURRENT STATUS
INTE	INTERRUPT ENABLE
CLK	CLOCK (INT F-F)
ELR	ENABLE LEVEL READ
ETLG	ENABLE THIS LEVEL GROUP
OUTPUTS:	
$\overline{A_0}-\overline{A_2}$	REQUEST LEVELS
INT	INTERRUPT (ACT. LOW) } OPEN COLLECTOR
ENLG	ENABLE NEXT LEVEL GROUP

### LOGIC DIAGRAM



## D.C. AND OPERATING CHARACTERISTICS

### ABSOLUTE MAXIMUM RATINGS\*

Temperature Under Bias	0°C to 70°C
Storage Temperature	-65°C to +150°C
All Output and Supply Voltages	-0.5V to +7V
All Input Voltages	-1.0V to +5.5V
Output Currents	100 mA

\*COMMENT: Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specifications is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$ .

Symbol	Parameter	Limits			Unit	Conditions
		Min.	Typ. <sup>(1)</sup>	Max.		
$V_C$	Input Clamp Voltage (all inputs)			-1.0	V	$I_C = -5\text{mA}$
$I_F$	Input Forward Current: ETLG input all other inputs		-.15 -.08	-0.5 -0.25	mA mA	$V_F = 0.45\text{V}$
$I_R$	Input Reverse Current: ETLG input all other inputs			80 40	$\mu\text{A}$ $\mu\text{A}$	$V_R = 5.25\text{V}$
$V_{IL}$	Input LOW Voltage: all inputs			0.8	V	$V_{CC} = 5.0\text{V}$
$V_{IH}$	Input HIGH Voltage: all inputs	2.0			V	$V_{CC} = 5.0\text{V}$
$I_{CC}$	Power Supply Current		90	130	mA	See Note 2.
$V_{OL}$	Output LOW Voltage: all outputs		.3	.45	V	$I_{OL} = 15\text{mA}$
$V_{OH}$	Output HIGH Voltage: ENLG output	2.4	3.0		V	$I_{OH} = -1\text{mA}$
$I_{OS}$	Short Circuit Output Current: ENLG output	-20	-35	-55	mA	$V_{OS} = 0\text{V}$ , $V_{CC} = 5.0\text{V}$
$I_{CEX}$	Output Leakage Current: $\overline{INT}$ and $\overline{A_0-A_2}$			100	$\mu\text{A}$	$V_{CEX} = 5.25\text{V}$

#### NOTES:

1. Typical values are for  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$ .
2.  $B_0-B_2$ ,  $\overline{SGS}$ , CLK,  $\overline{R_0-R_4}$  grounded, all other inputs and all outputs open.

**A.C. CHARACTERISTICS AND WAVEFORMS**  $T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$ 

Symbol	Parameter	Limits			Unit
		Min.	Typ. <sup>[1]</sup>	Max.	
$t_{CY}$	$\overline{CLK}$ Cycle Time	80	50		ns
$t_{PW}$	$\overline{CLK}$ , $\overline{ECS}$ , $\overline{INT}$ Pulse Width	25	15		ns
$t_{ISS}$	INTE Setup Time to $\overline{CLK}$	16	12		ns
$t_{ISH}$	INTE Hold Time after $\overline{CLK}$	20	10		ns
$t_{ETCS}^{[2]}$	ETLG Setup Time to $\overline{CLK}$	25	12		ns
$t_{ETCH}^{[2]}$	ETLG Hold Time After $\overline{CLK}$	20	10		ns
$t_{ECCS}^{[2]}$	$\overline{ECS}$ Setup Time to $\overline{CLK}$	80	50		ns
$t_{ECCH}^{[3]}$	$\overline{ECS}$ Hold Time After $\overline{CLK}$	0			ns
$t_{ECRS}^{[3]}$	$\overline{ECS}$ Setup Time to $\overline{CLK}$	110	70		ns
$t_{ECRH}^{[3]}$	$\overline{ECS}$ Hold Time After $\overline{CLK}$	0			
$t_{ECSS}^{[2]}$	$\overline{ECS}$ Setup Time to $\overline{CLK}$	75	70		ns
$t_{ECSH}^{[2]}$	$\overline{ECS}$ Hold Time After $\overline{CLK}$	0			ns
$t_{DCS}^{[2]}$	SGS and $\overline{B_0-B_2}$ Setup Time to $\overline{CLK}$	70	50		ns
$t_{DCH}^{[2]}$	SGS and $\overline{B_0-B_2}$ Hold Time After $\overline{CLK}$	0			ns
$t_{RCS}^{[3]}$	$\overline{R_0-R_7}$ Setup Time to $\overline{CLK}$	90	55		ns
$t_{RCH}^{[3]}$	$\overline{R_0-R_7}$ Hold Time After $\overline{CLK}$	0			ns
$t_{ICS}$	$\overline{INT}$ Setup Time to $\overline{CLK}$	55	35		ns
$t_{CI}$	$\overline{CLK}$ to $\overline{INT}$ Propagation Delay		15	25	ns
$t_{RIS}^{[4]}$	$\overline{R_0-R_7}$ Setup Time to $\overline{INT}$	10	0		ns
$t_{RIH}^{[4]}$	$\overline{R_0-R_7}$ Hold Time After $\overline{INT}$	35	20		ns
$t_{RA}$	$\overline{R_0-R_7}$ to $\overline{A_0-A_2}$ Propagation Delay		80	100	ns
$t_{ELA}$	$\overline{ELR}$ to $\overline{A_0-A_2}$ Propagation Delay		40	55	ns
$t_{ECA}$	$\overline{ECS}$ to $\overline{A_0-A_2}$ Propagation Delay		100	120	ns
$t_{ETA}$	ETLG to $\overline{A_0-A_2}$ Propagation Delay		35	70	ns
$t_{DECS}^{[4]}$	SGS and $\overline{B_0-B_2}$ Setup Time to $\overline{ECS}$	15	10		ns
$t_{DECH}^{[4]}$	SGS and $\overline{B_0-B_2}$ Hold Time After $\overline{ECS}$	15	10		ns
$t_{REN}$	$\overline{R_0-R_7}$ to ENLG Propagation Delay		45	70	ns
$t_{ETEN}$	ETLG to ENLG Propagation Delay		20	25	ns
$t_{ECRN}$	$\overline{ECS}$ to ENLG Propagation Delay		85	90	ns
$t_{ECSN}$	$\overline{ECS}$ to ENLG Propagation Delay		35	55	ns

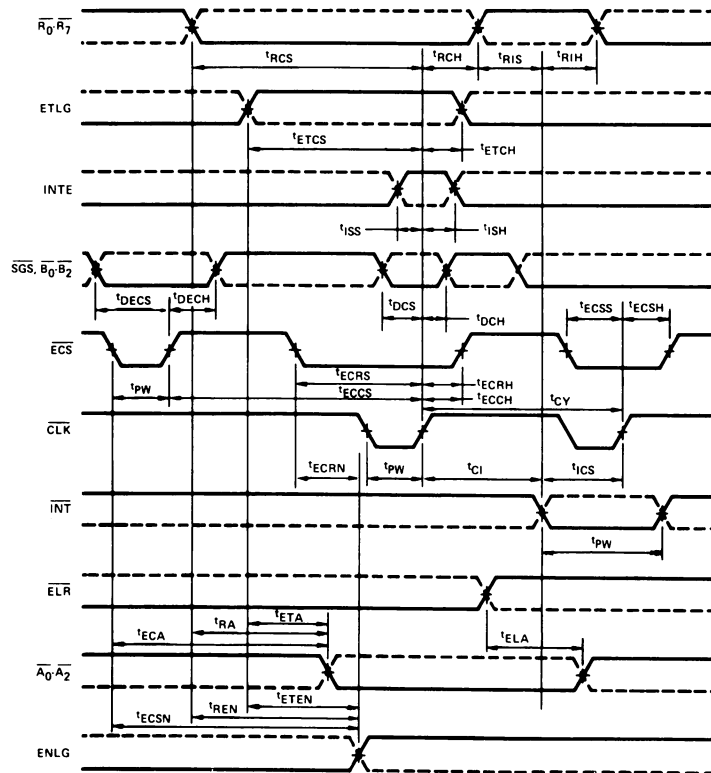
**CAPACITANCE [5]**

Symbol	Parameter	Limits			Unit
		Min.	Typ. <sup>[1]</sup>	Max	
$C_{IN}$	Input Capacitance		5	10	pF
$C_{OUT}$	Output Capacitance		7	12	pF

**TEST CONDITIONS:**  $V_{BIAS} = 2.5\text{V}$ ,  $V_{CC} = 5\text{V}$ ,  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{ MHz}$ 

NOTE 5. This parameter is periodically sampled and not 100% tested.

## WAVEFORMS



## NOTES:

- (1) Typical values are for  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$ .
- (2) Required for proper operation if ISE is enabled during next clock pulse.
- (3) These times are not required for proper operation but for desired change in interrupt flip-flop.
- (4) Required for new request or status to be properly loaded.

## TEST CONDITIONS:

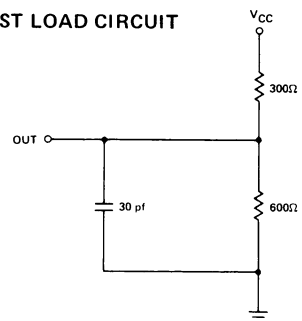
Input pulse amplitude: 2.5 volts.

Input rise and fall times: 5 ns between 1 and 2 volts.

Output loading of 15 mA and 30 pF.

Speed measurements taken at the 1.5V levels.

## TEST LOAD CIRCUIT



## 4 BIT PARALLEL BIDIRECTIONAL BUS DRIVER

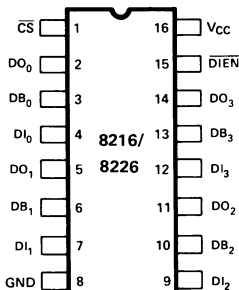
- Data Bus Buffer Driver for 8080 CPU
- Low Input Load Current — .25 mA Maximum
- High Output Drive Capability for Driving System Data Bus
- 3.65V Output High Voltage for Direct Interface to 8080 CPU
- Three State Outputs
- Reduces System Package Count

The 8216/8226 is a 4-bit bi-directional bus driver/receiver.

All inputs are low power TTL compatible. For driving MOS, the DO outputs provide a high 3.65V  $V_{OH}$ , and for high capacitance terminated bus structures, the DB outputs provide a high 50mA  $I_{OL}$  capability.

A non-inverting (8216) and an inverting (8226) are available to meet a wide variety of applications for buffering in micro-computer systems.

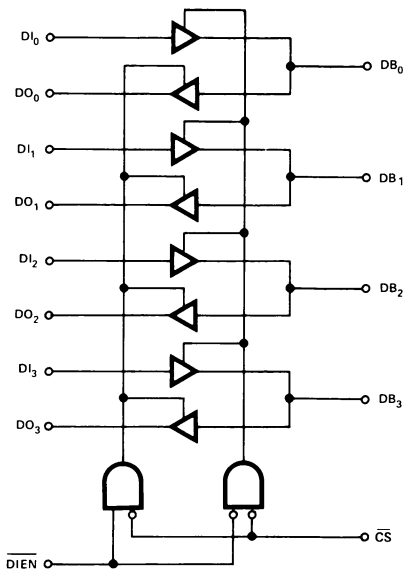
### PIN CONFIGURATION



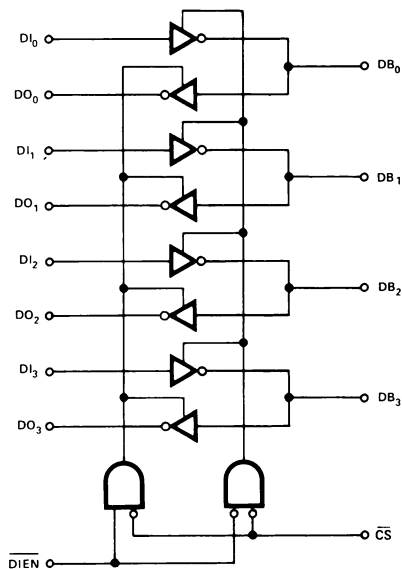
### PIN NAMES

DB <sub>0</sub> -DB <sub>3</sub>	DATA BUS BI-DIRECTIONAL
DI <sub>0</sub> -DI <sub>3</sub>	DATA INPUT
DO <sub>0</sub> -DO <sub>3</sub>	DATA OUTPUT
DIEN	DATA IN ENABLE DIRECTION CONTROL
CS	CHIP SELECT

### LOGIC DIAGRAM 8216



### LOGIC DIAGRAM 8226



## FUNCTIONAL DESCRIPTION

Microprocessors like the 8080 are MOS devices and are generally capable of driving a single TTL load. The same is true for MOS memory devices. While this type of drive is sufficient in small systems with few components, quite often it is necessary to buffer the microprocessor and memories when adding components or expanding to a multi-board system.

The 8216/8226 is a four bit bi-directional bus driver specifically designed to buffer microcomputer system components.

### Bi-Directional Driver

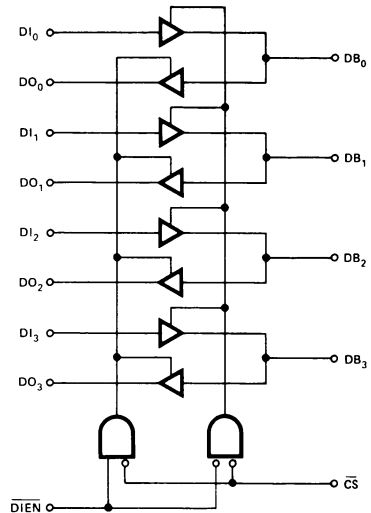
Each buffered line of the four bit driver consists of two separate buffers that are tri-state in nature to achieve direct bus interface and bi-directional capability. On one side of the driver the output of one buffer and the input of another are tied together (DB), this side is used to interface to the system side components such as memories, I/O, etc., because its interface is direct TTL compatible and it has high drive (50mA). On the other side of the driver the inputs and outputs are separated to provide maximum flexibility. Of course, they can be tied together so that the driver can be used to buffer a true bi-directional bus such as the 8080 Data Bus. The DO outputs on this side of the driver have a special high voltage output drive capability (3.65V) so that direct interface to the 8080 and 8008 CPUs is achieved with an adequate amount of noise immunity (350mV worst case).

### Control Gating $\overline{\text{DIEN}}$ , $\overline{\text{CS}}$

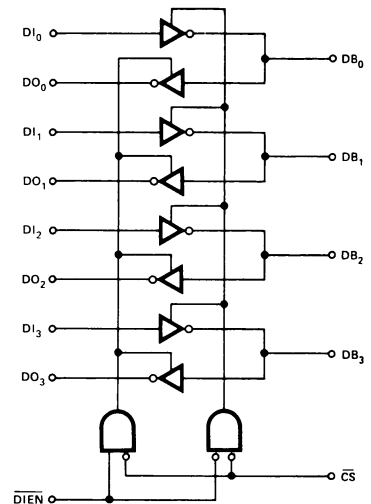
The  $\overline{\text{CS}}$  input is actually a device select. When it is "high" the output drivers are all forced to their high-impedance state. When it is at "zero" the device is selected (enabled) and the direction of the data flow is determined by the  $\overline{\text{DIEN}}$  input.

The  $\overline{\text{DIEN}}$  input controls the direction of data flow (see Figure 1) for complete truth table. This direction control is accomplished by forcing one of the pair of buffers into its high impedance state and allowing the other to transmit its data. A simple two gate circuit is used for this function.

The 8216/8226 is a device that will reduce component count in microcomputer systems and at the same time enhance noise immunity to assure reliable, high performance operation.



(a) 8216



(b) 8226

$\overline{\text{DIEN}}$	$\overline{\text{CS}}$	
0	0	DI $\rightarrow$ DB
1	0	DB $\rightarrow$ DO
0	1	HIGH IMPEDANCE
1	1	

Figure 1. 8216/8226 Logic Diagrams

## APPLICATIONS OF 8216/8226

## 8080 Data Bus Buffer

The 8080 CPU Data Bus is capable of driving a single TTL load and is more than adequate for small, single board systems. When expanding such a system to more than one board to increase I/O or Memory size, it is necessary to provide a buffer. The 8216/8226 is a device that is exactly fitted to this application.

Shown in Figure 2 are a pair of 8216/8226 connected directly to the 8080 Data Bus and associated control signals. The buffer is bi-directional in nature and serves to isolate the CPU data bus.

On the system side, the DB lines interface with standard semiconductor I/O and Memory components and are completely TTL compatible. The DB lines also provide a high drive capability (50mA) so that an extremely large system can be driven along with possible bus termination networks.

On the 8080 side the DI and DO lines are tied together and are directly connected to the 8080 Data Bus for bi-directional operation. The DO outputs of the 8216/8226 have a high voltage output capability of 3.65 volts which allows direct connection to the 8080 whose minimum input voltage is 3.3 volts. It also gives a very adequate noise margin of 350mV (worst case).

The  $\overline{\text{DIEN}}$  inputs to 8216/8226 is connected directly to the 8080.  $\overline{\text{DIEN}}$  is tied to DBIN so that proper bus flow is maintained, and  $\overline{\text{CS}}$  is tied to  $\overline{\text{BUSEN}}$  so that the system side Data Bus will be 3-stated when a Hold request has been acknowledged during a DMA activity.

## Memory and I/O Interface to a Bi-directional Bus

In large microcomputer systems it is often necessary to provide Memory and I/O with their own buffers and at the same time maintain a direct, common interface to a bi-directional Data Bus. The 8216/8226 has separated data in and data out lines on one side and a common bi-directional set on the other to accommodate such a function.

Shown in Figure 3 is an example of how the 8216/8226 is used in this type of application.

The interface to Memory is simple and direct. The memories used are typically Intel® 8102, 8102A, 8101 or 8107B-4 and have separate data inputs and outputs. The DI and DO lines of the 8216/8226 tie to them directly and under control of the  $\overline{\text{MEMR}}$  signal, which is connected to the  $\overline{\text{DIEN}}$  input, an interface to the bi-directional Data Bus is maintained.

The interface to I/O is similar to Memory. The I/O devices used are typically Intel® 8255s, and can be used for both input and output ports. The  $\overline{\text{I/O R}}$  signal is connected directly to the  $\overline{\text{DIEN}}$  input so that proper data flow from the I/O device to the Data Bus is maintained.

The 8216/8226 can be used in a wide variety of other buffering functions in microcomputer systems such as Address Bus Drivers, Drivers to peripheral devices such as printers, and as Drivers for long length cables to other peripherals or systems.

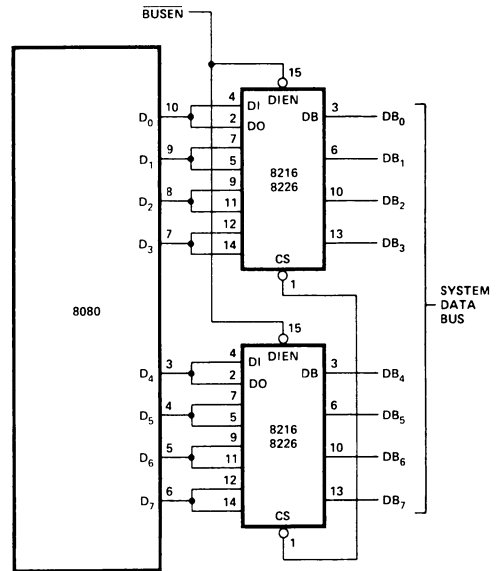


Figure 2. 8080 Data Bus Buffer.

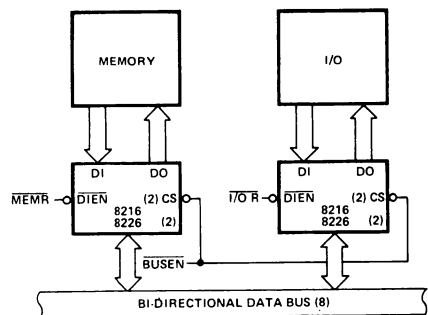


Figure 3. Memory and I/O Interface to a Bi-Directional Bus.

## D.C. AND OPERATING CHARACTERISTICS

## ABSOLUTE MAXIMUM RATINGS\*

Temperature Under Bias	0°C to 70°C
Storage Temperature	-65°C to +150°C
All Output and Supply Voltages	-0.5V to +7V
All Input Voltages	-1.0V to +5.5V
Output Currents	125 mA

\*COMMENT: Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied.

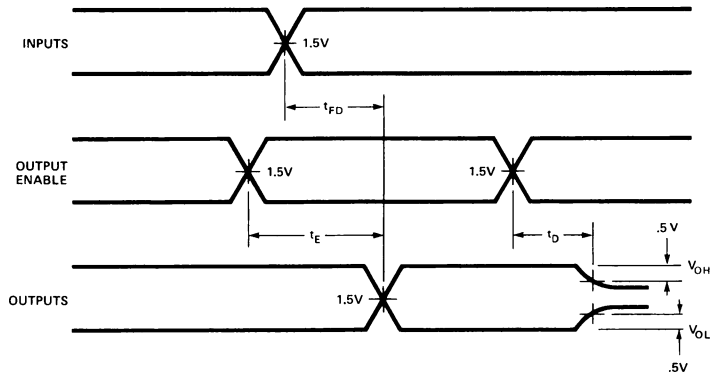
$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$

Symbol	Parameter	Limits			Unit	Conditions
		Min.	Typ.	Max.		
$I_{F1}$	Input Load Current $\overline{DIEN}$ , $\overline{CS}$		-0.15	-.5	mA	$V_F = 0.45$
$I_{F2}$	Input Load Current All Other Inputs		-0.08	-.25	mA	$V_F = 0.45$
$I_{R1}$	Input Leakage Current $\overline{DIEN}$ , $\overline{CS}$			20	$\mu\text{A}$	$V_R = 5.25\text{V}$
$I_{R2}$	Input Leakage Current DI Inputs			10	$\mu\text{A}$	$V_R = 5.25\text{V}$
$V_C$	Input Forward Voltage Clamp			-1	V	$I_C = -5\text{mA}$
$V_{IL}$	Input "Low" Voltage			.95	V	
$V_{IH}$	Input "High" Voltage	2.0			V	
$ I_O $	Output Leakage Current (3-State) DO DB			20 100	$\mu\text{A}$	$V_O = 0.45\text{V}/5.25\text{V}$
$I_{CC}$	Power Supply Current 8216		95	130	mA	
	8226		85	120	mA	
$V_{OL1}$	Output "Low" Voltage		0.3	.45	V	DO Outputs $I_{OL} = 15\text{mA}$ DB Outputs $I_{OL} = 25\text{mA}$
$V_{OL2}$	Output "Low" Voltage 8216		0.5	.6	V	DB Outputs $I_{OL} = 55\text{mA}$
	8226		0.5	.6	V	DB Outputs $I_{OL} = 50\text{mA}$
$V_{OH1}$	Output "High" Voltage	3.65	4.0		V	DO Outputs $I_{OH} = -1\text{mA}$
$V_{OH2}$	Output "High" Voltage	2.4	3.0		V	DB Outputs $I_{OH} = -10\text{mA}$
$I_{OS}$	Output Short Circuit Current	-15	-35	-65	mA	DO Outputs $V_O \cong 0\text{V}$ , DB Outputs $V_{CC} = 5.0\text{V}$
		-30	-75	-120	mA	

NOTE: Typical values are for  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$ .



## WAVEFORMS



## A.C. CHARACTERISTICS

$T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 5\%$

Symbol	Parameter	Limits			Unit	Conditions
		Min.	Typ. <sup>[1]</sup>	Max.		
$T_{PD1}$	Input to Output Delay DO Outputs		15	25	ns	$C_L = 30\text{pF}$ , $R_1 = 300\Omega$ $R_2 = 600\Omega$
$T_{PD2}$	Input to Output Delay DB Outputs 8216		20	30	ns	$C_L = 300\text{pF}$ , $R_1 = 90\Omega$ $R_2 = 180\Omega$
	8226		16	25	ns	
$T_E$	Output Enable Time 8216		45	65	ns	(Note 2)
	8226		35	54	ns	(Note 3)
$T_D$	Output Disable Time		20	35	ns	(Note 4)

## TEST CONDITIONS:

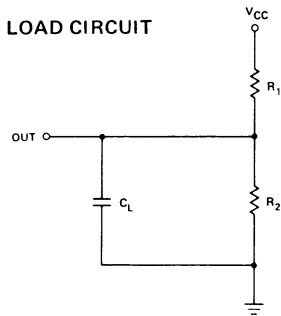
Input pulse amplitude of 2.5V.

Input rise and fall times of 5 ns between 1 and 2 volts.

Output loading is 5 mA and 10 pF.

Speed measurements are made at 1.5 volt levels.

## TEST LOAD CIRCUIT

Capacitance<sup>[5]</sup>

Symbol	Parameter	Limits			Unit
		Min.	Typ. <sup>[1]</sup>	Max.	
$C_{IN}$	Input Capacitance		4	8	pF
$C_{OUT1}$	Output Capacitance		6	10	pF
$C_{OUT2}$	Output Capacitance		13	18	pF

TEST CONDITIONS:  $V_{BIAS} = 2.5\text{V}$ ,  $V_{CC} = 5.0\text{V}$ ,  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{ MHz}$ .

- NOTES:
- Typical values are for  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$ .
  - DO Outputs,  $C_L = 30\text{pF}$ ,  $R_1 = 300/10\text{ K}\Omega$ ,  $R_2 = 180/1\text{ K}\Omega$ ; DB Outputs,  $C_L = 300\text{pF}$ ,  $R_1 = 90/10\text{ K}\Omega$ ,  $R_2 = 180/1\text{ K}\Omega$ .
  - DO Outputs,  $C_L = 30\text{pF}$ ,  $R_1 = 300/10\text{ K}\Omega$ ,  $R_2 = 600/1\text{ K}\Omega$ ; DB Outputs,  $C_L = 300\text{pF}$ ,  $R_1 = 90/10\text{ K}\Omega$ ,  $R_2 = 180/1\text{ K}\Omega$ .
  - DO Outputs,  $C_L = 5\text{pF}$ ,  $R_1 = 300/10\text{ K}\Omega$ ,  $R_2 = 600/1\text{ K}\Omega$ ; DB Outputs,  $C_L = 5\text{pF}$ ,  $R_1 = 90/10\text{ K}\Omega$ ,  $R_2 = 180/1\text{ K}\Omega$ .
  - This parameter is periodically sampled and not 100% tested.



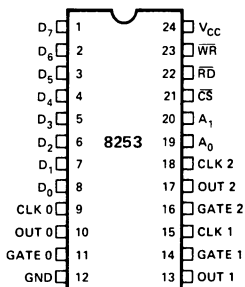
## 8253 PROGRAMMABLE INTERVAL TIMER

- 3 Independent 16-Bit Counters
- DC to 2 MHz
- Programmable Counter Modes
- Count Binary or BCD
- Single +5V Supply
- 24 Pin Dual-In-Line Package

The 8253 is a programmable counter/timer chip designed for use as an MCS-80™ peripheral. It uses nMOS technology with a single +5V supply and is packaged in a 24-pin plastic DIP.

It is organized as three independent 16-bit counters, each with a count rate of up to 2 MHz. All modes of operation are software programmable by the 8080.

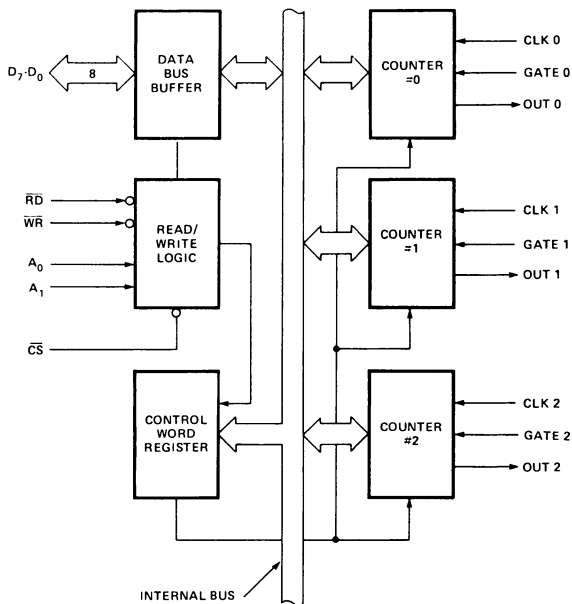
### PIN CONFIGURATION



### PIN NAMES

D <sub>7</sub> -D <sub>0</sub>	DATA BUS (8-BIT)
CLK N	COUNTER CLOCK INPUTS
GATE N	COUNTER GATE INPUTS
OUT N	COUNTER OUTPUTS
RD	READ COUNTER
WR	WRITE COMMAND OR DATA
CS	CHIP SELECT
A <sub>0</sub> , A <sub>1</sub>	COUNTER SELECT
V <sub>CC</sub>	+5 VOLTS
GND	GROUND

### BLOCK DIAGRAM



## 8253 BASIC FUNCTIONAL DESCRIPTION

### General

The 8253 is a programmable interval timer/counter specifically designed for use with the Intel® 8080 Microcomputer system. Its function is that of a general purpose, multi-mode timing element that can be treated as an array of I/O ports in the system software.

The 8253 solves one of the most common problems in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in systems software, the programmer configures the 8253 to match his requirements, initializes one of the counters of the 8253 with the desired quantity, then upon command the 8253 will count out the delay and interrupt the CPU when it has completed its tasks. It is easy to see that the software overhead is minimal and that multiple delays can easily be maintained by assignment of priority levels.

Other counter/timer functions that are non-delay in nature but also common to most microcomputers can be implemented with the 8253.

- Programmable Rate Generator
- Event Counter
- Binary Rate Multiplier
- Real Time Clock
- Digital One-Shot
- Complex Motor Controller

### Data Bus Buffer

This 3-state, bi-directional, 8-bit buffer is used to interface the 8253 to the MCS-80™ system data bus. Data is transmitted or received by the buffer upon execution of INput or OUTput CPU instructions. The Data Bus Buffer has three basic functions.

1. Programming the MODES of the 8253.
2. Loading the count registers.
3. Reading the count values.

### Read/Write Logic

The Read/Write Logic accepts inputs from the MCS-80™ system bus and in turn generates control signals for overall device operation. It is enabled or disabled by  $\overline{CS}$  so that no operation can occur to change the function unless the device has been selected by the system logic.

### $\overline{RD}$ (Read)

A "low" on this input informs the 8253 that the CPU is inputting data in the form of a counters value.

### $\overline{WR}$ (Write)

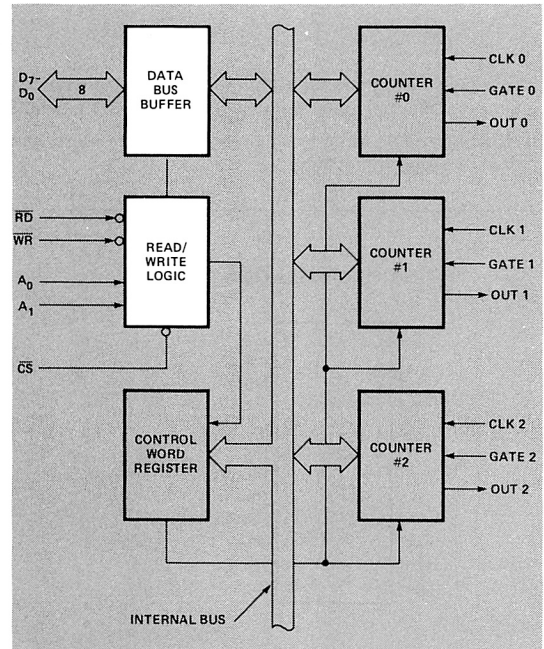
A "low" on this input informs the 8253 that the CPU is outputting data in the form of mode information or loading counters.

### A0, A1

These inputs are normally connected to the MCS-80™ address bus. Their function is to select one of the three counters to be operated on and to address the control word register for mode selection.

### $\overline{CS}$ (Chip Select)

A "low" on this input enables the 8253. No reading or writing will occur unless the device is selected. The  $\overline{CS}$  input has no effect upon the actual operation of the counters.



8253 BLOCK DIAGRAM

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A <sub>1</sub>	A <sub>0</sub>	
0	1	0	0	0	Load Counter No. 0
0	1	0	0	1	Load Counter No. 1
0	1	0	1	0	Load Counter No. 2
0	1	0	1	1	Write Mode Word
0	0	1	0	0	Read Counter No. 0
0	0	1	0	1	Read Counter No. 1
0	0	1	1	0	Read Counter No. 2
0	0	1	1	1	No-Operation 3-State
1	X	X	X	X	Disable 3-State
0	1	1	X	X	No-Operation 3-State

### Control Word Register

The Control Word Register is selected when A0, A1 are 11. It then accepts information from the data bus buffer and stores it in a register. The information stored in this register controls the operational MODE of each counter, selection of binary or BCD counting and the loading of each count register.

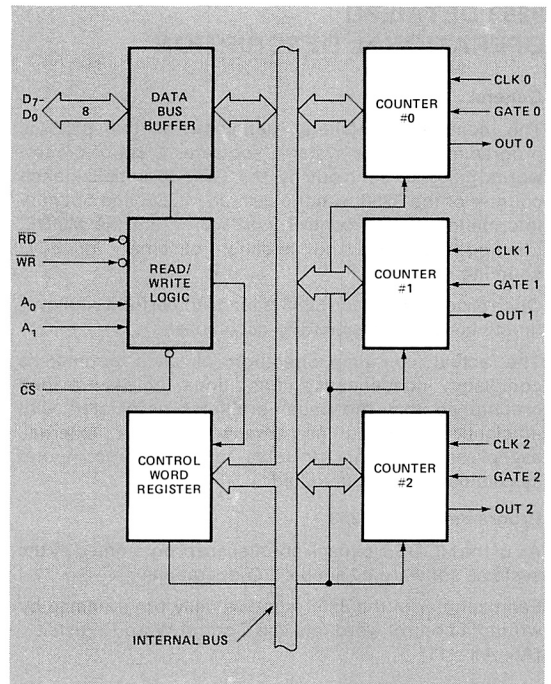
The Control Word Register can only be written into; no read operation of its contents is available.

### Counter #0, Counter #1, Counter #2

These three functional blocks are identical in operation so only a single Counter will be described. Each Counter consists of a single, 16-bit, pre-settable, DOWN counter. The counter can operate in either binary or BCD and its input, gate and output are configured by the selection of MODES stored in the Control Word Register.

The counters are fully independent and each can have separate Mode configuration and counting operation, binary or BCD. Also, there are special features in the control word that handle the loading of the count value so that software overhead can be minimized for these functions.

The reading of the contents of each counter is available to the programmer with simple READ operations for event counting applications and special commands and logic are included in the 8253 so that the contents of each counter can be read "on the fly" without having to inhibit the clock input.

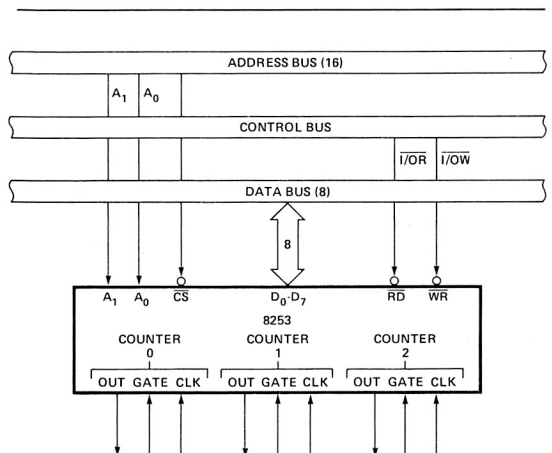


8253 BLOCK DIAGRAM

### 8253 SYSTEM INTERFACE

The 8253 is a component of the Intel<sup>®</sup> MCS-80 System and interfaces in the same manner as all other peripherals of the family. It is treated by the systems software as an array of peripheral I/O ports; three are counters and the fourth is a control register for MODE programming.

Basically, the select inputs A0, A1 connect to the A0, A1 address bus signals of the CPU. The  $\overline{CS}$  can be derived directly from the address bus using a linear select method. Or it can be connected to the output of a decoder, such as an Intel<sup>®</sup> 8205 for larger systems. The  $\overline{RD}$  and  $\overline{WR}$  inputs are normally connected to the  $\overline{IOR}$  and  $\overline{IOW}$  outputs of the 8228 but they can be connected to the  $\overline{MEMR}$  and  $\overline{MEMW}$  signals in a memory mapped I/O configuration so that the full memory operating instructions of the 8080A can be used to initialize and maintain the 8253.



8253 SYSTEM INTERFACE

## 8253 DETAILED OPERATIONAL DESCRIPTION

### General

The complete functional definition of the 8253 is programmed by the systems software. A set of control words must be sent out by the CPU to initialize each counter of the 8253 with the desired MODE and quantity information. These control words program the MODE, Loading sequence and selection of binary or BCD counting.

Once programmed, the 8253 is ready to perform whatever timing tasks it is assigned to accomplish.

The actual counting operation of each counter is completely independent and additional logic is provided on-chip so that the usual problems associated with efficient monitoring and management of external, asynchronous events or rates to the microcomputer system have been eliminated.

### Programming the 8253

All of the MODES for each counter are programmed by the systems software by simple I/O operations.

Each counter of the 8253 is individually programmed by writing a control word into the Control Word Register. (A0, A1 = 11)

### Control Word Format

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

### Definition of Control Fields

#### SC-Select Counter

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

#### RL-Read/Load

RL1	RL0	
0	0	Counter Latching operation (see READ/WRITE Procedure Section)
1	0	Read/Load most significant byte only.
0	1	Read/Load least significant byte only.
1	1	Read/Load least significant byte first, then most significant byte.

### M-MODE

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

### BCD

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

### MODE Definition

#### MODE 0: Interrupt on terminal count.

The OUTput will be initially low after the Mode set operation. After the count is loaded into the selected count register, the OUTput will remain low and the counter will count. When terminal count is reached the OUTput will go high and remain high until the selected count register is reloaded with the Mode.

Reloading a counter register during counting results in the following:

- (1) Load 1st byte stops the current counting.
- (2) Load 2nd byte starts the new count.

The GATE input will enable the counting when high and inhibit counting when low.

#### MODE 1: Programmable One-Shot.

The OUTput will go low on the count following the rising edge of the GATE input.

The OUTput will go high on the terminal count. If a new count value is loaded while the OUTput is low it will not affect the duration of the One-Shot pulse until the succeeding trigger. The current count can be read at any time without affecting the one-shot pulse.

The one-shot is retriggerable, hence the output will remain low for the full count after any rising edge of the gate input.

**MODE 2: Rate Generator**

Divide by N counter. The OUTput will be low for one period of the input clock. The period from one output pulse to the next equals the number of input counts in the count register. If the count register is reloaded between output pulses the present period will not be affected, but the subsequent period will reflect the new value.

The GATE input, when low, will force the OUTput high. When the GATE input goes high, the counter will start from the initial count. Thus, the GATE input can be used to synchronize the counter.

When this MODE is set, the output will remain high until after the count register is loaded. The output then can also be synchronized by software.

**MODE 3: Square Wave Rate Generator.**

Similar to MODE 2 except that the OUTput will remain high until one half the count has been completed (for even numbers) and go low for the other half of the count. If the count is odd, the OUTput will be high for  $(N+1)/2$  counts and low for  $(N-1)/2$  counts.

If the counter register is reloaded with a new value during counting, this new value will be reflected immediately after the output transition of the current count.

**MODE 4: Software triggered strobe.**

After the mode is set, the output will be high. When the count is loaded, the counter will begin counting. On terminal count, the output will go low for one input clock period, then will go high again.

If the count register is reloaded between output pulses the present period will not be affected, but the subsequent period will reflect the new value. The count will be inhibited while the gate input is low. Reloading the counter register will restart counting beginning with the new number.

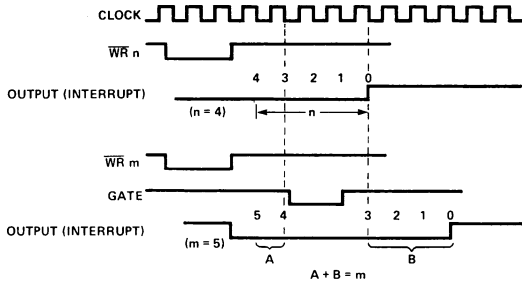
**MODE 5: Hardware triggered strobe.**

The counter will start counting after the rising edge of the trigger input and will go low for one clock period when the terminal count is reached. The counter is retriggerable. The output will not go low until the full count after the rising edge of any trigger.

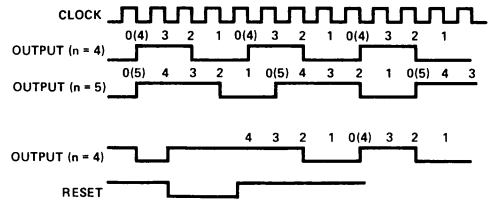
**GATE Pin Operations Summary**

Signal Status Modes	Low Or Going Low	Rising	High
0	Disables counting	---	Enables counting
1	---	1) Initiates counting 2) Resets output after next clock	---
2	1) Disables counting 2) Sets output immediately high	Initiates counting	Enables counting
3	1) Disables counting 2) Sets output immediately high	Initiates counting	Enables counting
4	Disables counting	---	Enables counting
5	---	Initiates counting	---

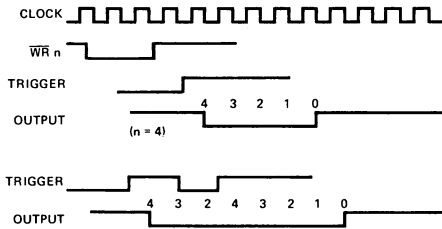
## MODE 0



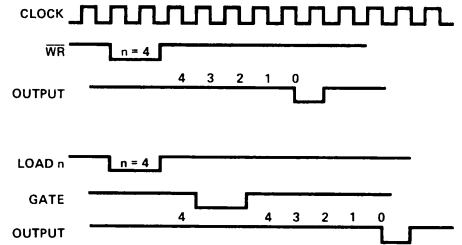
## MODE 3



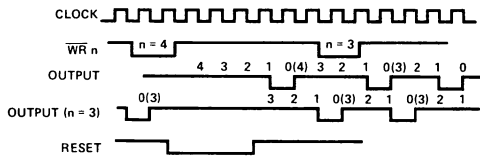
## MODE 1



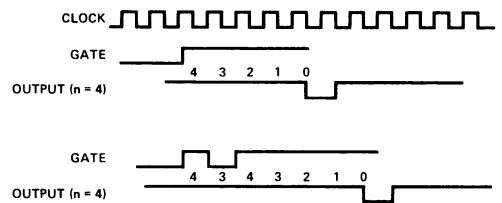
## MODE 4



## MODE 2



## MODE 5



## 8253 TIMING DIAGRAMS



## 8253 READ/WRITE PROCEDURE

### Write Operations

The systems software must program each counter of the 8253 with the mode and quantity desired. The programmer must write out to the 8253 a MODE control word and the programmed number of count register bytes (1 or 2) prior to actually using the selected counter.

The actual order of the programming is quite flexible. Writing out of the MODE control word can be in any sequence of counter selection, e.g., counter #0 does not have to be first or counter #2 last. Each counter's MODE control word register has a separate address so that its loading is completely sequence independent. (SC0, SC1)

The loading of the Count Register with the actual count value, however, must be done in exactly the sequence programmed in the MODE control word (RL0, RL1). This loading of the counter's count register is still sequence independent like the MODE control word loading, but when a selected count register is to be loaded it must be loaded with the number of bytes programmed in the MODE control word (RL0, RL1). The one or two bytes to be loaded in the count register do not have to follow the associated MODE control word. They can be programmed at any time following the MODE control word loading as long as the correct number of bytes is loaded in order.

All counters are down counters. Thus, the value loaded into the count register will actually be decremented. Loading all zeroes into a count register will result in the maximum count ( $2^{16}$  for Binary or  $10^4$  for BCD). In MODE 0 the new count will not restart until the load has been completed. It will accept one of two bytes depending on how the MODE control words (RL0, RL1) are programmed. Then proceed with the restart operation.

### Programming Format

MODE Control Word Counter n	
LSB	Count Register byte Counter n
MSB	Count Register byte Counter n

Note: Format shown is a simple example of loading the 8253 and does not imply that it is the only format that can be used.

### Alternate Programming Formats

Example:

		A1	A0
No. 1	MODE Control Word Counter 0	1	1
No. 2	MODE Control Word Counter 1	1	1
No. 3	MODE Control Word Counter 2	1	1
No. 4	LSB Count Register Byte Counter 1	0	1
No. 5	MSB Count Register Byte Counter 1	0	1
No. 6	LSB Count Register Byte Counter 2	1	0
No. 7	MSB Count Register Byte Counter 2	1	0
No. 8	LSB Count Register Byte Counter 0	0	0
No. 9	MSB Count Register Byte Counter 0	0	0

Note: The exclusive addresses of each counter's count register make the task of programming the 8253 a very simple matter, and maximum effective use of the device will result if this feature is fully utilized.

## 8253 READ/WRITE PROCEDURE

### Read Operations

In most counter applications it becomes necessary to read the value of the count in progress and make a computational decision based on this quantity. Event counters are probably the most common application that uses this function. The 8253 contains logic that will allow the programmer to easily read the contents of any of the three counters without disturbing the actual count in progress.

There are two methods that the programmer can use to read the value of the counters. The first method involves the use of simple I/O read operations of the selected counter. By controlling the A0, A1 inputs to the 8253 the programmer can select the counter to be read (remember that no read operation of the mode register is allowed A0, A1-11). The only requirement with this method is that in order to assure a stable count reading the actual operation of the selected counter must be inhibited either by controlling the Gate input or by external logic that inhibits the clock input. The contents of the counter selected will be available as follows:

first I/O Read contains the least significant byte (LSB).

second I/O Read contains the most significant byte (MSB).

Due to the internal logic of the 8253 it is absolutely necessary to complete the entire reading procedure. If two bytes are programmed to be read then two bytes must be read before any loading WR command can be sent to the same counter.

### Read Operation Chart

A1	A0	RD	
0	0	0	Read Counter No. 0
0	1	0	Read Counter No. 1
1	0	0	Read Counter No. 2
1	1	0	Illegal

### Reading While Counting

In order for the programmer to read the contents of any counter without effecting or disturbing the counting operation the 8253 has special internal logic that can be accessed using simple WR commands to the MODE register. Basically, when the programmer wishes to read the contents of a selected counter "on the fly" he loads the MODE register with a special code which latches the present count value into a storage register so that its contents contain an accurate, stable quantity. The programmer then issues a normal read command to the selected counter and the contents of the latched register is available.

### MODE Register for Latching Count

A0, A1 = 11

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	0	0	X	X	X	X

SC1, SC0 — specify counter to be latched.

D5, D4 — 00 designates counter latching operation.

X — don't care.

The same limitation applies to this mode of reading the counter as the previous method. That is, it is mandatory to complete the entire read operation as programmed.

## Absolute Maximum Ratings

Ambient Temperature Under Bias ..... 0° C to 70° C  
 Storage Temperature ..... -65° C to +150° C  
 Voltage On Any Pin  
     With Respect to Ground ..... -0.5 V to +7 V  
 Power Dissipation ..... 1 Watt

**\*COMMENT:**

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

## D.C. Characteristics: ( $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ ; $V_{CC} = 5\text{V} \pm 5\%$ )

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$V_{IL}$	Input Low Voltage	-.5	.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC} + .5\text{V}$	V	
$V_{OL}$	Output Low Voltage		.45	V	$I_{OL} = 2\text{ mA}$
$V_{OH}$	Output High Voltage	2.4		V	$I_{OH} = -400\text{ }\mu\text{A}$
$I_{LI}$	Input Load Current		10	$\mu\text{A}$	$V_{IN} = V_{CC}$ to 0V
$I_{LOL}$	Output Leakage Current		-10	$\mu\text{A}$	$V_{OUT} = 0.45\text{V}$
$I_{LOH}$	Output Leakage Current		10	$\mu\text{A}$	$V_{OUT} = V_{CC}$
$I_{CC}$	$V_{CC}$ Supply Current		85	mA	

## Capacitance $T_A = 25^\circ\text{C}$ ; $V_{CC} = \text{GND} = 0\text{V}$

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
$C_{IN}$	Input Capacitance			10	pF	$f_c = 1\text{ MHz}$ Unmeasured pins returned to $V_{SS}$
$C_{I/O}$	I/O Capacitance			20	pF	

**A.C. Characteristics:**  $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ ;  $V_{CC} = 5.0\text{V} \pm 5\%$ ;  $\text{GND} = 0\text{V}$

**BUS PARAMETERS:** (Note 1)

**READ CYCLE**

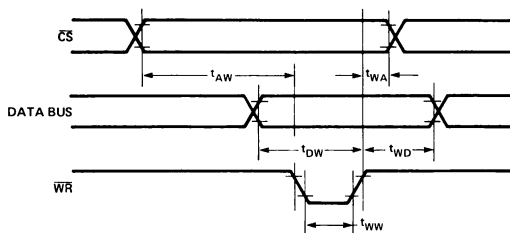
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AR}$	Address Stable Before $\overline{\text{READ}}$	50		ns	
$t_{RA}$	Address Hold Time for $\overline{\text{READ}}$	5		ns	
$t_{RR}$	$\overline{\text{READ}}$ Pulse Width	430		ns	
$t_{RD}$	Data Delay from $\overline{\text{READ}}$		350	ns	$C_L = 100\text{ pF}$
$t_{DF}$	$\overline{\text{READ}}$ to Data Floating		200	ns	$C_L = 100\text{ pF}$
		25		ns	$C_L = 15\text{ pF}$

**WRITE CYCLE**

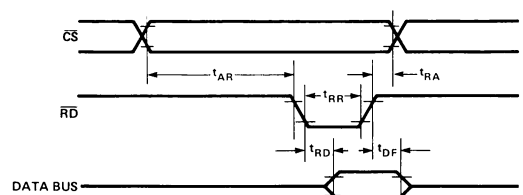
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AW}$	Address Stable Before $\overline{\text{WRITE}}$	20		ns	
$t_{WA}$	Address Hold Time for $\overline{\text{WRITE}}$	20		ns	
$t_{WW}$	$\overline{\text{WRITE}}$ Pulse Width	400		ns	
$t_{DW}$	Data Set Up Time for $\overline{\text{WRITE}}$	200		ns	
$t_{WD}$	Data Hold Time for $\overline{\text{WRITE}}$	40		ns	
$t_{RV}$	Recovery Time Between $\overline{\text{WRITES}}$	1		$\mu\text{s}$	

Note 1: AC timings measured at  $V_{OH} = 2.0$ ,  $V_{OL} = .8$ , and with load circuit of Figure 1.

**WRITE TIMING**



**READ TIMING**

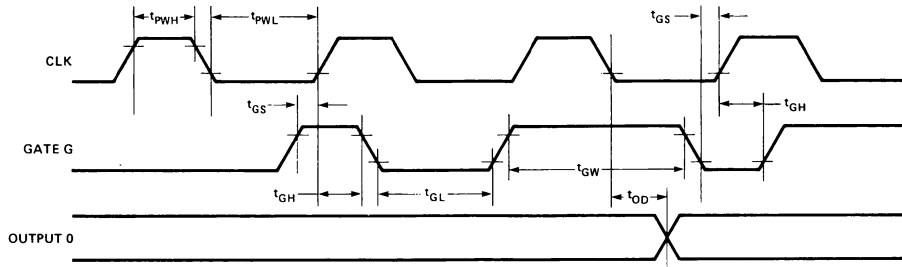


**PRELIMINARY**  
 Warning: This is not a final specification. Some  
 characteristics limits are subject to change.

**A.C. CHARACTERISTICS (Cont'd):**  $T_A = 0^\circ\text{C to } 70^\circ\text{C}$ ;  $V_{CC} = 5.0\text{V} \pm 5\%$ ;  $GND = 0\text{V}$

### CLOCK AND GATE TIMING

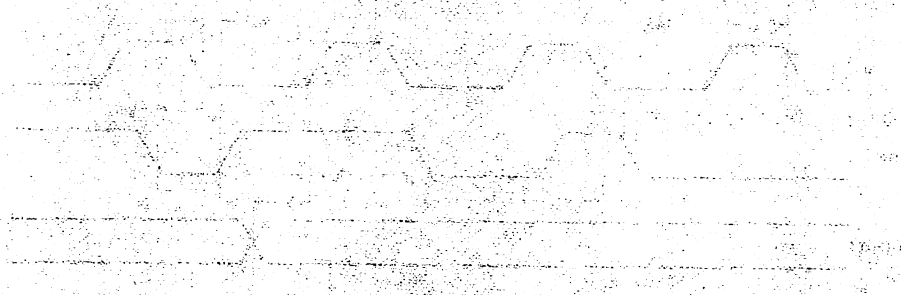
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{CLK}$	Clock Period	300	dc	ns	
$t_{PWH}$	High Pulse Width	200		ns	
$t_{PWL}$	Low Pulse Width	100		ns	
$t_{GW}$	Trigger Pulse Width	200		ns	
$t_{GS}$	Gate Set Up Time To CLK $\uparrow$	150		ns	
$t_{GH}$	Gate Hold Time After CLK $\uparrow$	100		ns	
$t_{GL}$	Low Gate Width	100		ns	
$t_{OD}$	Output Delay From CLK $\downarrow$		300	ns	$C_L = 50\text{ pF}$



JOINT AIR FORCE AND NAVY JOINT MILITARY ACADEMY

JOINT STAGION MODEL

STATION	TIME	STATION	TIME	STATION	TIME
1	0000	2	0000	3	0000
4	0000	5	0000	6	0000
7	0000	8	0000	9	0000
10	0000	11	0000	12	0000
13	0000	14	0000	15	0000
16	0000	17	0000	18	0000
19	0000	20	0000	21	0000
22	0000	23	0000	24	0000
25	0000	26	0000	27	0000
28	0000	29	0000	30	0000
31	0000	32	0000	33	0000
34	0000	35	0000	36	0000
37	0000	38	0000	39	0000
40	0000	41	0000	42	0000
43	0000	44	0000	45	0000
46	0000	47	0000	48	0000
49	0000	50	0000	51	0000
52	0000	53	0000	54	0000
55	0000	56	0000	57	0000
58	0000	59	0000	60	0000
61	0000	62	0000	63	0000
64	0000	65	0000	66	0000
67	0000	68	0000	69	0000
70	0000	71	0000	72	0000
73	0000	74	0000	75	0000
76	0000	77	0000	78	0000
79	0000	80	0000	81	0000
82	0000	83	0000	84	0000
85	0000	86	0000	87	0000
88	0000	89	0000	90	0000
91	0000	92	0000	93	0000
94	0000	95	0000	96	0000
97	0000	98	0000	99	0000
100	0000	101	0000	102	0000
103	0000	104	0000	105	0000
106	0000	107	0000	108	0000
109	0000	110	0000	111	0000
112	0000	113	0000	114	0000
115	0000	116	0000	117	0000
118	0000	119	0000	120	0000
121	0000	122	0000	123	0000
124	0000	125	0000	126	0000
127	0000	128	0000	129	0000
130	0000	131	0000	132	0000
133	0000	134	0000	135	0000
136	0000	137	0000	138	0000
139	0000	140	0000	141	0000
142	0000	143	0000	144	0000
145	0000	146	0000	147	0000
148	0000	149	0000	150	0000
151	0000	152	0000	153	0000
154	0000	155	0000	156	0000
157	0000	158	0000	159	0000
160	0000	161	0000	162	0000
163	0000	164	0000	165	0000
166	0000	167	0000	168	0000
169	0000	170	0000	171	0000
172	0000	173	0000	174	0000
175	0000	176	0000	177	0000
178	0000	179	0000	180	0000
181	0000	182	0000	183	0000
184	0000	185	0000	186	0000
187	0000	188	0000	189	0000
190	0000	191	0000	192	0000
193	0000	194	0000	195	0000
196	0000	197	0000	198	0000
199	0000	200	0000	201	0000
202	0000	203	0000	204	0000
205	0000	206	0000	207	0000
208	0000	209	0000	210	0000
211	0000	212	0000	213	0000
214	0000	215	0000	216	0000
217	0000	218	0000	219	0000
220	0000	221	0000	222	0000
223	0000	224	0000	225	0000
226	0000	227	0000	228	0000
229	0000	230	0000	231	0000
232	0000	233	0000	234	0000
235	0000	236	0000	237	0000
238	0000	239	0000	240	0000
241	0000	242	0000	243	0000
244	0000	245	0000	246	0000
247	0000	248	0000	249	0000
250	0000	251	0000	252	0000
253	0000	254	0000	255	0000
256	0000	257	0000	258	0000
259	0000	260	0000	261	0000
262	0000	263	0000	264	0000
265	0000	266	0000	267	0000
268	0000	269	0000	270	0000
271	0000	272	0000	273	0000
274	0000	275	0000	276	0000
277	0000	278	0000	279	0000
280	0000	281	0000	282	0000
283	0000	284	0000	285	0000
286	0000	287	0000	288	0000
289	0000	290	0000	291	0000
292	0000	293	0000	294	0000
295	0000	296	0000	297	0000
298	0000	299	0000	300	0000
301	0000	302	0000	303	0000
304	0000	305	0000	306	0000
307	0000	308	0000	309	0000
310	0000	311	0000	312	0000
313	0000	314	0000	315	0000
316	0000	317	0000	318	0000
319	0000	320	0000	321	0000
322	0000	323	0000	324	0000
325	0000	326	0000	327	0000
328	0000	329	0000	330	0000
331	0000	332	0000	333	0000
334	0000	335	0000	336	0000
337	0000	338	0000	339	0000
340	0000	341	0000	342	0000
343	0000	344	0000	345	0000
346	0000	347	0000	348	0000
349	0000	350	0000	351	0000
352	0000	353	0000	354	0000
355	0000	356	0000	357	0000
358	0000	359	0000	360	0000
361	0000	362	0000	363	0000
364	0000	365	0000	366	0000
367	0000	368	0000	369	0000
370	0000	371	0000	372	0000
373	0000	374	0000	375	0000
376	0000	377	0000	378	0000
379	0000	380	0000	381	0000
382	0000	383	0000	384	0000
385	0000	386	0000	387	0000
388	0000	389	0000	390	0000
391	0000	392	0000	393	0000
394	0000	395	0000	396	0000
397	0000	398	0000	399	0000
400	0000	401	0000	402	0000
403	0000	404	0000	405	0000
406	0000	407	0000	408	0000
409	0000	410	0000	411	0000
412	0000	413	0000	414	0000
415	0000	416	0000	417	0000
418	0000	419	0000	420	0000
421	0000	422	0000	423	0000
424	0000	425	0000	426	0000
427	0000	428	0000	429	0000
430	0000	431	0000	432	0000
433	0000	434	0000	435	0000
436	0000	437	0000	438	0000
439	0000	440	0000	441	0000
442	0000	443	0000	444	0000
445	0000	446	0000	447	0000
448	0000	449	0000	450	0000
451	0000	452	0000	453	0000
454	0000	455	0000	456	0000
457	0000	458	0000	459	0000
460	0000	461	0000	462	0000
463	0000	464	0000	465	0000
466	0000	467	0000	468	0000
469	0000	470	0000	471	0000
472	0000	473	0000	474	0000
475	0000	476	0000	477	0000
478	0000	479	0000	480	0000
481	0000	482	0000	483	0000
484	0000	485	0000	486	0000
487	0000	488	0000	489	0000
490	0000	491	0000	492	0000
493	0000	494	0000	495	0000
496	0000	497	0000	498	0000
499	0000	500	0000	501	0000
502	0000	503	0000	504	0000
505	0000	506	0000	507	0000
508	0000	509	0000	510	0000
511	0000	512	0000	513	0000
514	0000	515	0000	516	0000
517	0000	518	0000	519	0000
520	0000	521	0000	522	0000
523	0000	524	0000	525	0000
526	0000	527	0000	528	0000
529	0000	530	0000	531	0000
532	0000	533	0000	534	0000
535	0000	536	0000	537	0000
538	0000	539	0000	540	0000
541	0000	542	0000	543	0000
544	0000	545	0000	546	0000
547	0000	548	0000	549	0000
550	0000	551	0000	552	0000
553	0000	554	0000	555	0000
556	0000	557	0000	558	0000
559	0000	560	0000	561	0000
562	0000	563	0000	564	0000
565	0000	566	0000	567	0000
568	0000	569	0000	569	0000



## 8259

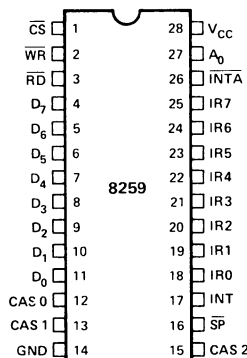
# PROGRAMMABLE INTERRUPT CONTROLLER

- Eight Level Priority Controller
- Expandable to 64 Levels
- Programmable Interrupt Modes (Algorithms)
- Individual Request Mask Capability
- Single +5V Supply (No Clocks)
- 28 Pin Dual-In-Line Package
- Fully Compatible with 8080 CPU

The 8259 handles up to eight vectored priority interrupts for the 8080A CPU. It is cascadable for up to 64 vectored priority interrupts, without additional circuitry. It will be packaged in a 28-pin plastic DIP, uses nMOS technology and requires a single +5V supply. Circuitry is static, requiring no clock input.

The 8259 is designed to minimize the software and real time overhead in handling multi-level priority interrupts. It has several modes, permitting optimization for a variety of system requirements.

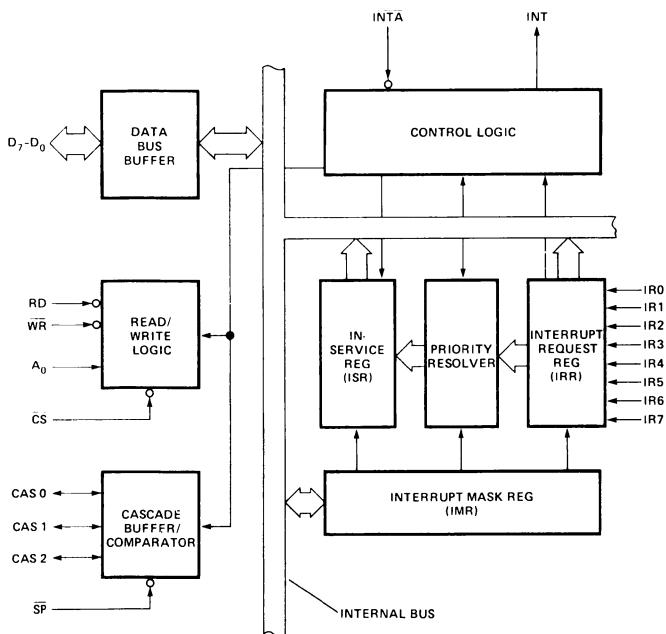
### PIN CONFIGURATION



### PIN NAMES

D <sub>7</sub> -D <sub>0</sub>	DATA BUS (BI-DIRECTIONAL)
RD	READ INPUT
WR	WRITE INPUT
A <sub>0</sub>	COMMAND SELECT ADDRESS
CS	CHIP SELECT
CAS1-CAS0	CASCADE LINES
SP	SLAVE PROGRAM INPUT
INT	INTERRUPT OUTPUT
INTA	INTERRUPT ACKNOWLEDGE INPUT
IR0-IR7	INTERRUPT REQUEST INPUTS

### BLOCK DIAGRAM



## INTERRUPTS IN MICROCOMPUTER SYSTEMS

Microcomputer system design requires that I/O devices such as keyboards, displays, sensors and other components receive servicing in an efficient method so that large amounts of the total system tasks can be assumed by the microcomputer with little or no effect on throughput.

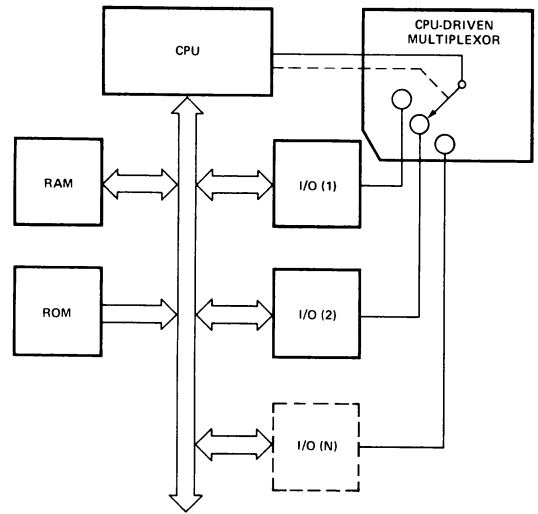
The most common method of servicing such devices is the **Polled** approach. This is where the processor must test each device in sequence and in effect "ask" each one if it needs servicing. It is easy to see that a large portion of the main program is looping through this continuence polling cycle and that such a method would have a serious, detrimental effect on system throughput thus limiting the tasks that could be assumed by the microcomputer and reducing the cost effectiveness of using such devices.

A more desireable method would be one that would allow the microprocessor to be executing its main program and only stop to service peripheral devices when it is told to do so by the device itself. In effect, the method would provide an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is complete however the processor would resume exactly where it left off.

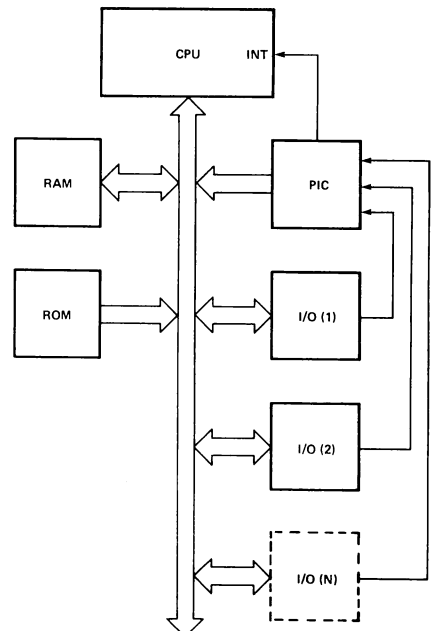
This method is called **Interrupt**. It is easy to see that system throughput would drastically increase, and thus more tasks could be assumed by the microcomputer to further enhance its cost effectiveness.

The Programmable Interrupt Controller (PIC) functions as an overall manager in an Interrupt-Driven system environment. It accepts requests from the peripheral equipment, determines which of the incoming requests is of the highest importance (priority), ascertains whether the incoming request has a higher priority value than the level currently being serviced and issues an Interrupt to the CPU based on this determination.

Each peripheral device or structure usually has a special program or "routine" that is associated with its specific functional or operational requirements; this is referred to as a "service routine". The PIC, after issuing an Interrupt to the CPU, must somehow input information into the CPU that can "point" the Program Counter to the service routine associated with the requesting device. The PIC does this by providing the CPU with a 3-byte CALL instruction.



### POLLED METHOD



### INTERRUPT METHOD



## 8259 BASIC FUNCTIONAL DESCRIPTION

### General

The 8259 is a device specifically designed for use in real time, interrupt driven, microcomputer systems. It manages eight levels or requests and has built-in features for expandability to other 8259s (up to 64 levels). It is programmed by the system's software as an I/O peripheral. A selection of priority algorithms is available to the programmer so that the manner in which the requests are processed by the 8259 can be configured to match his system requirements. The priority assignments and algorithms can be changed or reconfigured dynamically at any time during the main program. This means that the complete interrupt structure can be defined as required, based on the total system environment.

### Interrupt Request Register (IRR) and In-Service Register (ISR)

The interrupts at the IR input lines are handled by two registers in cascade, the Interrupt Request Register (IRR) and the In-Service Register (ISR). The IRR is used to store all the interrupt levels which are requesting service; and the ISR is used to store all the interrupt levels which are being serviced.

The IRR bit is set and INT line is raised high whenever there is a positive going edge at the IR input. However, the IR input must be held high until the 1st  $\overline{INTA}$  pulse has arrived. More than one bit of the IRR can be set at once as long as they are not masked. The IRR is reset by the  $\overline{INTA}$  sequence.

The ISR bit is set by the  $\overline{INTA}$  pulse (at the same time the selected IRR bit is reset). This bit remains set during the subroutine until an EOI (End of Interrupt) command is received by the 8259.

The return from the subroutine to the main program may look like this:

```
DI
OUT  OCW2  (Send EOI command)
POP  PSW
EI
RET
```

### Priority Resolver

This logic block determines the priorities of the bits set in the IRR. The highest priority is selected and strobed into the corresponding bit of the ISR during  $\overline{INTA}$  pulse.

### INT (Interrupt)

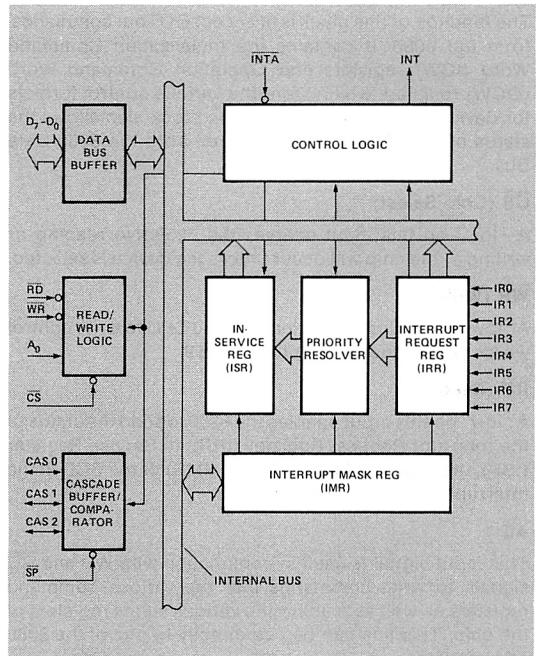
This output goes directly to the 8080 INT input. The  $V_{OH}$  level on this line is designed to be fully compatible with the 8080 input level.

### $\overline{INTA}$ (Interrupt Acknowledge)

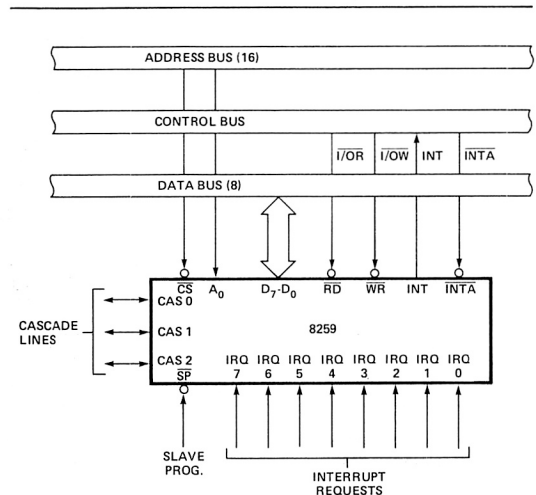
This input generally comes from the 8228 of the CPU group. The 8228 will produce 3 distinct  $\overline{INTA}$  pulses. The 3  $\overline{INTA}$  pulses will cause the 8259 to release a 3-byte CALL instruction onto the Data Bus.

### Interrupt Mask Register (IMR)

The IMR stores the bits of the interrupt lines to be masked. The IMR operates on both the IRR and the ISR. Masking of a higher priority bit will not affect the interrupt request lines of lower priority.



8259 BLOCK DIAGRAM



8259 INTERFACE TO 8080 STANDARD SYSTEM BUS

### Data Bus Buffer

This 3-state, bi-directional, 8-bit buffer is used to interface the 8259 to the 8080 system Data Bus. Control words and status information are transferred through the Data Bus Buffer.

### Read/Write Control Logic

The function of this block is to accept OUTPut commands from the 8080. It contains the Initialization Command Word (ICW) registers and Operation Command Word (OCW) registers which store the various control formats for device operation. This function block also allows the status of the 8259 to be transferred onto the 8080 Data Bus.

### $\overline{CS}$ (Chip Select)

A "low" on this input enables the 8259. No reading or writing of the chip will occur unless the device is selected.

### $\overline{WR}$ (Write)

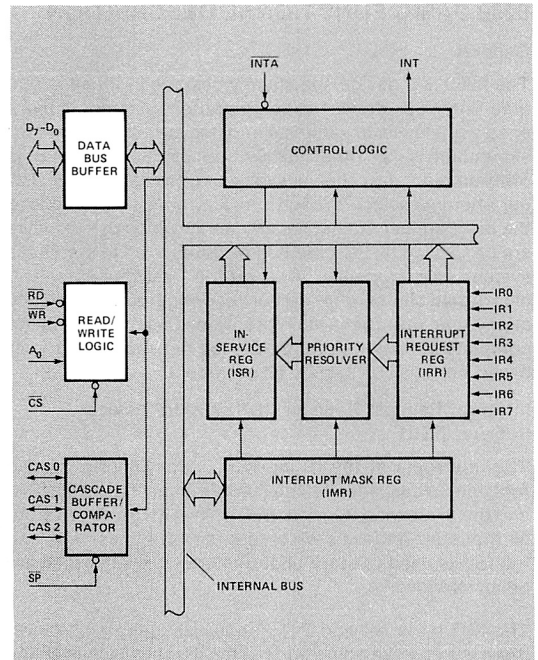
A "low" on this input enables the 8080 CPU to write control words (ICWs and OCWs) to the 8259.

### $\overline{RD}$ (Read)

A "low" on this input enables the 8259 to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR) or the BCD of the Interrupt level on to the Data Bus.

### $A_0$

This input signal is used in conjunction with  $\overline{WR}$  and  $\overline{RD}$  signals to write commands into the various command registers as well as reading the various status registers of the chip. This line can be tied directly to one of the 8080 address lines.



8259 BLOCK DIAGRAM

### 8259 BASIC OPERATION

$A_0$	$D_4$	$D_3$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0			0	1	0	IRR, ISR or Interrupting Level $\Rightarrow$ DATA BUS (Note 1)
1			0	1	0	IMR $\Rightarrow$ DATA BUS
OUTPUT OPERATION (WRITE)						
0	0	0	1	0	0	DATA BUS $\Rightarrow$ OCW2
0	0	1	1	0	0	DATA BUS $\Rightarrow$ OCW3
0	1	X	1	0	0	DATA BUS $\Rightarrow$ ICW1
1	X	X	1	0	0	DATA BUS $\Rightarrow$ OCW1, ICW2, ICW3 (Note 2)
DISABLE FUNCTION						
X	X	X	1	1	0	DATA BUS $\Rightarrow$ 3-STATE
X	X	X	X	X	1	DATA BUS $\Rightarrow$ 3-STATE

Note 1: Selection of IRR, ISR or Interrupting Level is based on the content of OCW3 written before the READ operation.

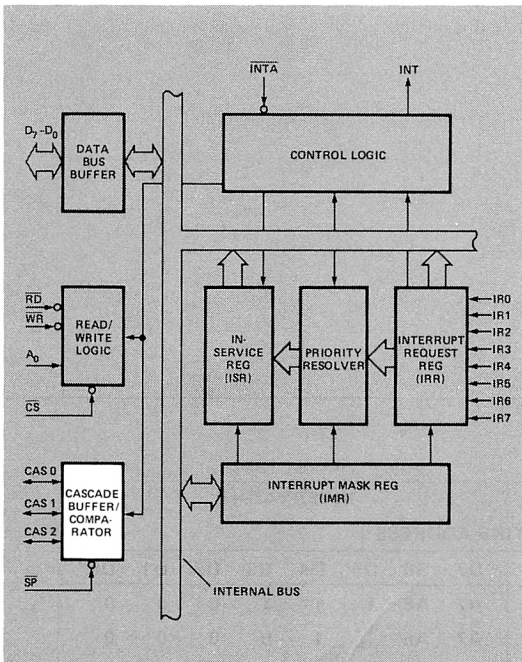
Note 2: On-chip sequencer logic queues these commands into proper sequence.

### SP (Slave Program)

More than one 8259 can be used in the system to expand the priority interrupt scheme up to 64 levels. In such case, one 8259 acts as the master, and the others act as slaves. A "high" on the  $\overline{SP}$  pin designates the 8259 as the master, a "low" designates it as a slave.

### The Cascade Buffer/Comparator

This function block stores and compares the IDs of all 8259 used in the system. The associated three I/O pins (CAS0-2) are outputs when the 8259 is used as a master ( $\overline{SP} = 1$ ), and are inputs when the 8259 is used as a slave ( $\overline{SP} = 0$ ). As a master, the 8259 sends the ID of the interrupting slave device onto the CAS0-2 lines. The slave thus selected will send its preprogrammed subroutine address onto the Data Bus during next two consecutive  $\overline{INTA}$  pulses. (See section "Cascading the 8259".)



8259 BLOCK DIAGRAM

## 8259 DETAILED OPERATIONAL SUMMARY

### General

The powerful features of the 8259 in the 8080 micro-computer system are its programmability and its utilization of the 8080 CALL instruction to jump into any address in the memory map. The normal sequence of events that the 8259 interacts with the CPU is as follows:

1. One or more of the INTERRUPT REQUEST lines ( $IR_7-0$ ) are raised high signaling the 8259 that the peripheral equipment(s) are demanding service.
2. The 8259 accepts these requests, resolves the priorities, and sends an  $INT$  to the 8080 CPU.

3. The 8080 CPU acknowledges the  $INT$  and responds with an  $\overline{INTA}$  pulse.
4. Upon receiving the  $\overline{INTA}$  from the CPU group (8228), the 8259 will release a CALL instruction code (11001101) onto the 8-bit Data Bus through its  $D_7-0$  pins.
5. This CALL instruction will initiate two more  $\overline{INTA}$  pulses to be sent to the 8259 from the CPU group (8228).
6. These two  $\overline{INTA}$  pulses allow the 8259 to release its preprogrammed subroutine address onto the Data Bus. The lower 8-bit address is released at the first  $\overline{INTA}$  pulse and the higher 8-bit address is released at the second  $\overline{INTA}$  pulse.
7. This completes the 3-byte CALL instruction released by the 8259. The In-Service Register (ISR) is not reset until the end of the subroutine when an EOI (End of interrupt) command is issued to the 8259.

### Programming The 8259

The 8259 accepts two types of command words generated by the CPU:

1. Initialization Command Words (ICWs):  
Before normal operation can begin, each 8259 in the system must be brought to a starting point — by a sequence of 2 or 3 bytes timed by  $\overline{WR}$  pulses. This sequence is described in Figure 1.
2. Operation Command Words (OCWs):  
These are the command words which command the 8259 to operate in various interrupt modes. These modes are:
  - a. Fully nested mode
  - b. Rotating priority mode
  - c. Special mask mode
  - d. Polled mode

The OCWs can be written into the 8259 at anytime during operation.

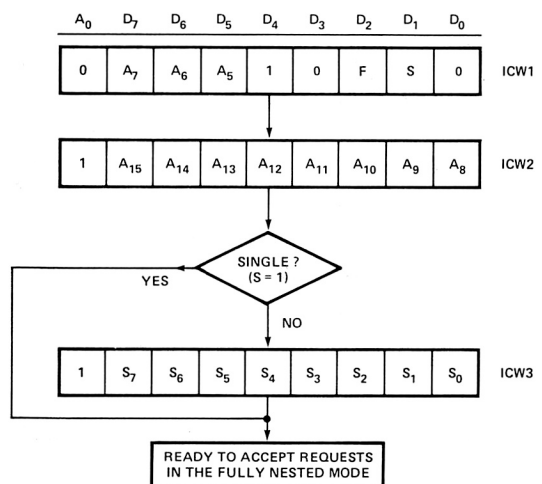


FIGURE 1. INITIALIZATION SEQUENCE

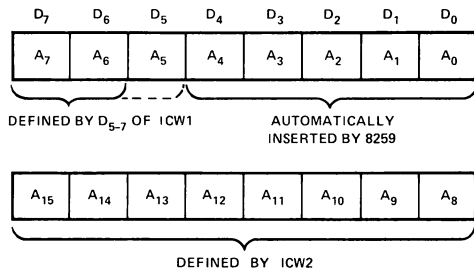
### Initialization Command Words 1 and 2: (ICW1 and ICW2)

Whenever a command is issued with A0 = 0 and D4 = 1, this is interpreted as Initialization Command Word 1 (ICW1), and initiates the initialization sequence. During this sequence, the following occur automatically:

- The edge sense circuit is reset, which means that following initialization, an interrupt request (IR) input must make a low to high transition to generate an interrupt.
- The interrupt Mask Register is cleared.
- IR 7 input is assigned priority 7.
- Special Mask Mode Flip-flop and status Read Flip-flop are reset.

The 8 requesting devices have 8 addresses equally spaced in memory. The addresses can be programmed at intervals of 4 or 8 bytes; the 8 routines thus occupying a page of 32 or 64 bytes respectively in memory.

The address format is:



A0-4 are automatically inserted by the 8259, while A15-6 are programmed by ICW1 and ICW2. When interval = 8, A5 is fixed by the 8259. If interval = 4, A5 is programmed in ICW1. Thus, the interrupt service routines can be located anywhere in the memory space. The 8 byte interval will maintain compatibility with current 8080 RESTART instruction software, while the 4 byte interval is best for compact jump table.

The address format inserted by the 8259 is described in Table 1.

The bits F and S are defined by ICW1 as follows:

F: Call address interval. F = 1, then interval = 4; F = 0, then interval = 8.

S: Single. S = 1 means that this is the only 8259 in the system. It avoids the necessity of programming ICW3.

INTERVAL = 4									INTERVAL = 8							
LOWER MEMORY ROUTINE ADDRESS																
	D7	D6	D5	D4	D3	D2	D1	D0	D7	D6	D5	D4	D3	D2	D1	D0
IR 7	A7	A6	A5	1	1	1	0	0	A7	A6	1	1	1	0	0	0
IR 6	A7	A6	A5	1	1	0	0	0	A7	A6	1	1	0	0	0	0
IR 5	A7	A6	A5	1	0	1	0	0	A7	A6	1	0	1	0	0	0
IR 4	A7	A6	A5	1	0	0	0	0	A7	A6	1	0	0	0	0	0
IR 3	A7	A6	A5	0	1	1	0	0	A7	A6	0	1	1	0	0	0
IR 2	A7	A6	A5	0	1	0	0	0	A7	A6	0	1	0	0	0	0
IR 1	A7	A6	A5	0	0	1	0	0	A7	A6	0	0	1	0	0	0
IR 0	A7	A6	A5	0	0	0	0	0	A7	A6	0	0	0	0	0	0

TABLE 1.

### Example of Interrupt Acknowledge Sequence

Assume the 8259 is programmed with  $F = 1$  (CALL address interval = 4), and  $IR_5$  is the interrupting level. The 3 byte sequence released by the 8259 timed by the  $\overline{INTA}$  pulses is as follows:

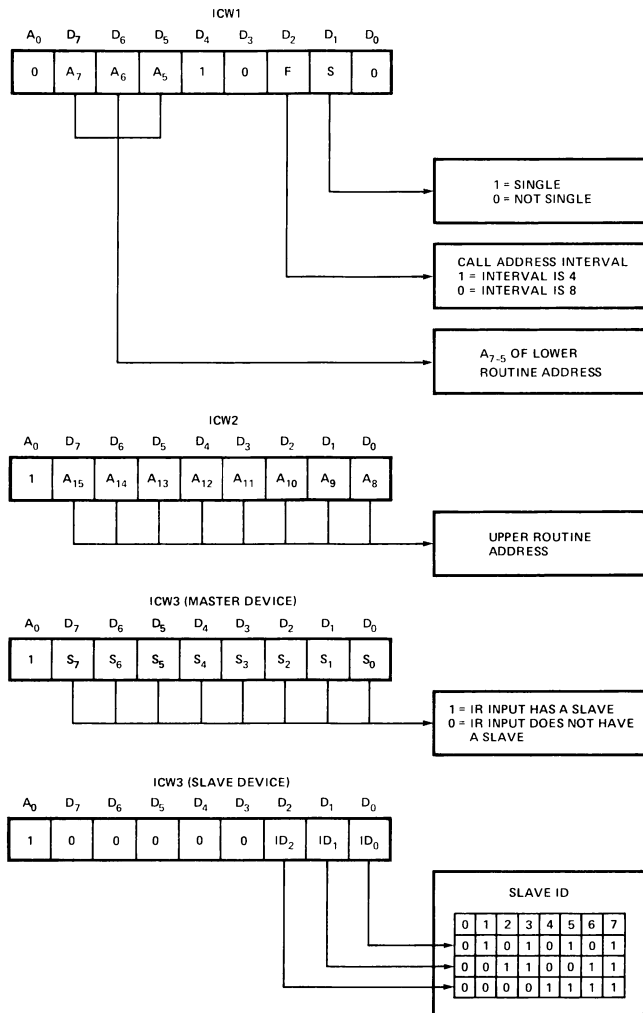
	D7	D6	D5	D4	D3	D2	D1	D0	
1st $\overline{INTA}$	1	1	0	0	1	1	0	1	CALL CODE
2nd $\overline{INTA}$	A7	A6	A5	1	0	1	0	0	LOWER ROUTINE ADDRESS
3rd $\overline{INTA}$	A15	A14	A13	A12	A11	A10	A9	A8	HIGHER ROUTINE ADDRESS

### Initialization Command Word 3 (ICW3)

This will load the 8-bit slave register. The functions of this register are as follows:

- If the 8259 is the master, a "1" is set for each slave in the system. The master then will release byte 1 of the CALL sequence and will enable the corresponding slave to release bytes 2 and 3, through the cascade lines.
- If the 8259 is a slave, bits 2 - 0 identify the slave. The slave compares its  $CAS_0-2$  inputs (sent by the master) with these bits. If they are equal, bytes 2 and 3 of the CALL sequence are released.

If bit S is set in ICW1, there is no need to program ICW3.



### Operation Command Words (OCWs)

After the Initialization Command Words (ICWs) are programmed into the 8259, the chip is ready to accept interrupt requests at its input lines. However, during the 8259 operation, a selection of algorithms can command the 8259 to operate in various modes through the Operation Command Words (OCWs). These various modes and their associated OCWs are described below.

#### Interrupt Masks

Each Interrupt Request input can be masked individually by the Interrupt Masked Register (IMR) programmed through OCW1.

The IMR will operate on both the Interrupt Request Register and the In-Service Register. Note that if an interrupt is already acknowledged by the 8259 (an INTA pulse has occurred), then the Interrupting level, although masked, will inhibit the lower priorities. To enable these lower priority interrupts, one can do one of the two things: (1) Write an End of Interrupt (EOI) command (OCW2) to reset the ISR bit or (2) Set the special mask mode using OCW3 (as will be explained later in the special mask mode.)

#### Fully Nested Mode

The 8259 will operate in the fully nested mode after the execution of the initialization sequence without any OCW being written. In this mode, the interrupt requests are ordered in priorities from 0 through 7. When an interrupt is acknowledged, the highest priority request is determined and its address vector placed on the bus. In addition, a bit of the Interrupt service register (IS 7-0) is set. This bit remains set until the 8080 issues an End of Interrupt (EOI) command immediately before returning from the service routine. While the IS bit is set, all further interrupts of lower priority are inhibited, while higher levels will be able to generate an interrupt (which will only be acknowledged if the 8080 has enabled its own interrupt input through software).

After the Initialization sequence, IR0 has the highest priority and IR7 the lowest. Priorities can be changed, as will be explained in the rotating priority mode.

#### Rotating Priority Modes

The Rotating Priority Modes of the 8259 serves in application of interrupting devices of equal priority such as communication channels. There are two variations of the rotating priority mode: the auto mode and the specific mode.

1. Auto Mode — In this mode, a device after being serviced receives the lowest priority, so a device requesting an interrupt will have to wait, in the worst case, until 7 other devices are serviced at most once each. i.e., if the priority and "in service" status is:

BEFORE ROTATE	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
"IS" STATUS	0	1	0	1	0	0	0	0
	LOWEST PRIORITY				HIGHEST PRIORITY			
PRIORITY STATUS	7	6	5	4	3	2	1	0

AFTER ROTATE	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
"IS" STATUS	0	1	0	0	0	0	0	0
	LOWEST PRIORITY				HIGHEST PRIORITY			
PRIORITY STATUS	4	3	2	1	0	7	6	5

In this example, the In-Service FF corresponding to line 4 (the highest priority FF set) was reset and line 4 became the lowest priority, while all the other priorities rotated correspondingly.

The Rotate command is issued in OCW2, where: R = 1, EOI = 1, SEOI = 0.

2. Specific Mode — The programmer can change priorities by programming the bottom priority, and by doing this, to fix the highest priority: i.e., if IR5 is programmed as the bottom priority device, the IR6 will have the highest one.

The Rotate command is issued in OCW2 where: R = 1, SEOI = 1. L2, L1, L0 are the BCD priority level codes of the bottom priority device.

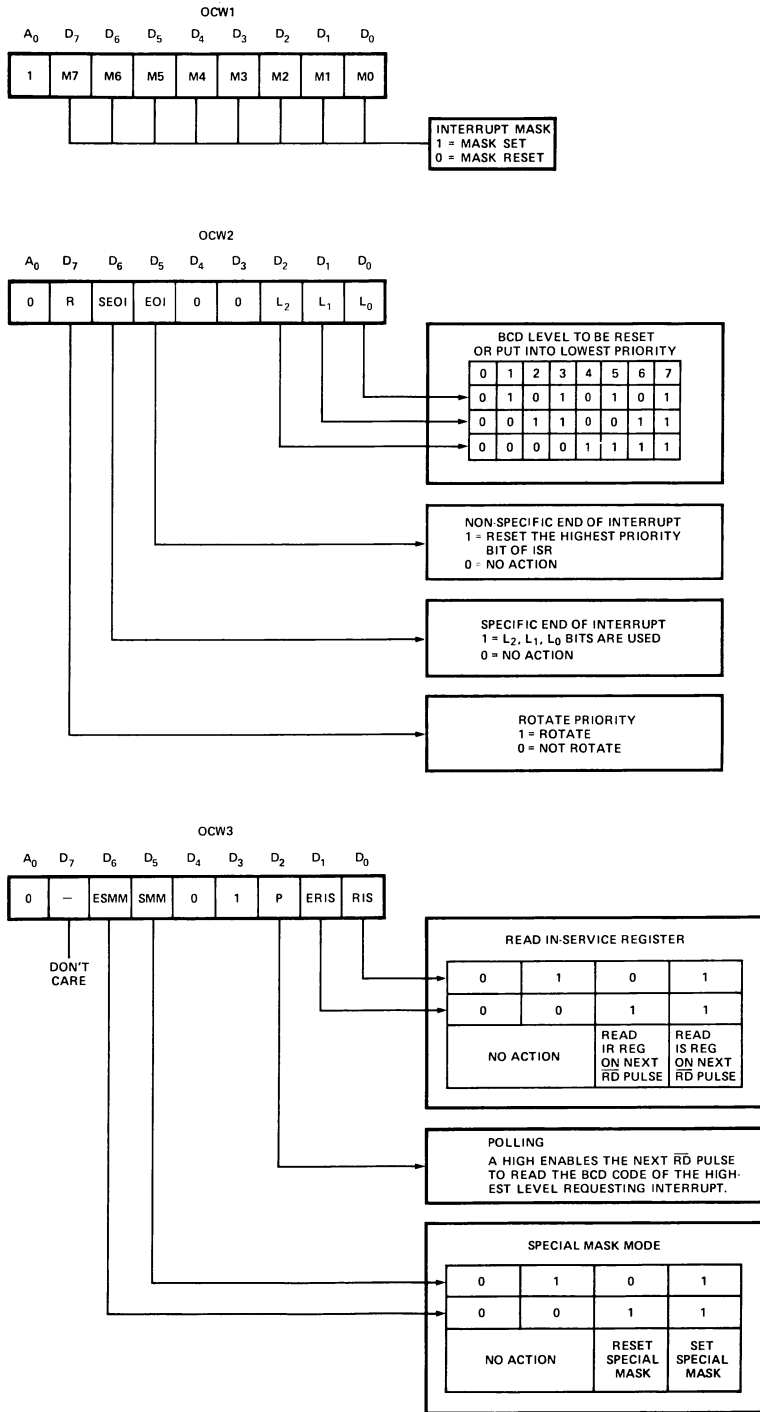
Observe that this mode is independent of the End of Interrupt Command and priority changes can be executed during EOI command or independently from the EOI command.

#### End of Interrupt (EOI) and Specific End of Interrupt (SEOI)

An End of Interrupt command word must be issued to the 8259 before returning from a service routine, to reset the appropriate IS bit.

There are two forms of EOI command: Specific and non-Specific. When the 8259 is operated in modes which preserve the fully nested structure, it can determine which IS bit to reset on EOI. When a non-Specific EOI command is issued the 8259 will automatically reset the highest IS bit of those that are set, since in the nested mode, the highest IS level was necessarily the last level acknowledged and will necessarily be the next routine level returned from.

However, when a mode is used which may disturb the fully nested structure, such as in the rotating priority case, the 8259 may no longer be able to determine the last level acknowledged. In this case, a specific EOI (SEOI) must be issued which includes the IS level to be reset as part of the command. The End of the Interrupt is issued whenever EOI = "1" in OCW2. For specific EOI, SEOI = "1", and EOI = 1. L2, L1, L0 is then the BCD level to be reset. As explained in the Rotate Mode earlier, this can also be the bottom priority code. Note that although the Rotate command can be issued during an EOI = 1, it is not necessarily tied to it.



### Special Mask Mode (SMM)

This mode is useful when some bit(s) are set (masked) by the Interrupt Mask Register (IMR) through OCW1. If, for some reason, we are currently in a subroutine which is masked (this could happen when the subroutine intentionally masks itself off). It is still possible to enable the lower priority lines by setting the Special Mask mode. In this mode the lower priority lines are enabled until the SMM is reset. The higher priorities are not affected.

The special mask mode FF is set by OCW3 where ESMM = 1, SMM = 1, and reset where: ESSM = 1 and SMM = 0.

### Polled Mode

In this mode, the 8080 disables its interrupt input. Service to devices is achieved by programmer initiative by a Poll command.

The poll command is issued by setting P = "1" in OCW3 during a  $\overline{WR}$  pulse.

The 8259 treats the next  $\overline{RD}$  pulse as an interrupt acknowledge, sets the appropriate IS Flip-flop, if there is a request, and reads the priority level.

The word enabled onto the data bus during  $\overline{RD}$  is:

D7	D6	D5	D4	D3	D2	D1	D0
I	-	-	-	-	W2	W1	W0

W0 — 2: BCD code of the highest priority level requesting service.

I: Equal to a "1" if there is an interrupt.

This mode is useful if there is a routine command common to several levels — so that the  $\overline{INTA}$  sequence is not needed (and this saves ROM space). Another application is to use the poll mode to expand the number of priority levels to more than 64.

## SUMMARY OF OPERATION COMMAND WORD PROGRAMMING

	A0	D4	D3		
OCW1	1			M7-M0	IMR (Interrupt Mask Register). $\overline{WR}$ will load it while status can be read with $\overline{RD}$ .
OCW2	0	0	0	R SEOI EOI 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	No Action. Non-specific End of Interrupt. No Action. Specific End of Interrupt. L2, L1, L0 is the BCD level to be reset. No Action. Rotate priority at EOI. (Auto Mode) Rotate priority, L2, L1, L0 becomes bottom priority without Ending of Interrupt. Rotate priority at EOI (Specific Mode), L2, L1, L0 becomes bottom priority, and its corresponding IS FF is reset.
OCW3	0	1	0	ESMM SMM 0 0 0 1 1 0 1 1 ERIS RIS 0 0 0 1 1 0 1 1	} Special Mask not Affected. Reset Special Mask. Set Special Mask. } No Action. Read IR Register Status. Read IS Register Status.

Note: The 8080 INT input must be disabled during:

1. Initialization sequence for all the 8259 in the system.
2. Any control command execution.



Polling overrides status read when P = 1, ERIS = 1 in OCW3.

The cascade bus lines are normally low and will contain the slave address code from the trailing edge of the first  $\overline{\text{INTA}}$  pulse to the trailing edge of the third pulse. It is obvious that each 8259 in the system must follow a separate initialization sequence and can be programmed to work in a different mode. An EOI command must be issued twice: once for the master and once for the corresponding slave. An address decoder is required to activate the Chip Select ( $\overline{\text{CS}}$ ) input of each 8259. The slave program pin ( $\overline{\text{SP}}$ ) must be at a "low" level for a slave (and then the cascade lines are inputs) and at a "high" level for a master (and then the cascade lines are outputs).



## 8259 INSTRUCTION SET

INST. NO.	MNEMONIC	A0	D7	D6	D5	D4	D3	D2	D1	D0	OPERATION DESCRIPTION
1	ICW1 A	0	A7	A6	A5	1	0	1	1	0	Byte 1 initialization, format = 4, single.
2	ICW1 B	0	A7	A6	A5	1	0	1	0	0	Byte 1 initialization, format = 4, not single.
3	ICW1 C	0	A7	A6	A5	1	0	0	1	0	Byte 1 initialization, format = 8, single.
4	ICW1 D	0	A7	A6	A5	1	0	0	0	0	Byte 1 initialization, format = 8, not single.
5	ICW2	1	A15	A14	A13	A12	A11	A10	A9	A8	Byte 2 initialization (Address No. 2)
6	ICW3 M	1	S7	S6	S5	S4	S3	S2	S1	S0	Byte 3 initialization — master.
7	ICW3 S	1	0	0	0	0	0	S2	S1	S0	Byte 3 initialization — slave.
8	OCW1	1	M7	M6	M5	M4	M3	M2	M1	M0	Load mask reg, read mask reg.
9	OCW2 E	0	0	0	1	0	0	0	0	0	Non specific EOI.
10	OCW2 SE	0	0	1	1	0	0	L2	L1	L0	Specific EOI. L2, L1, L0 code of IS FF to be reset.
11	OCW2 RE	0	1	0	1	0	0	0	0	0	Rotate at EOI (Auto Mode).
12	OCW2 RSE	0	1	1	1	0	0	L2	L1	L0	Rotate at EOI (Specific Mode). L2, L1, L0, code of line to be reset and selected as bottom priority.
13	OCW2 RS	0	1	1	0	0	0	L2	L1	L0	L2, L1, L0 code of bottom priority line.
14	OCW3 P	0	—	0	0	0	1	1	0	0	Poll mode.
15	OCW3 RIS	0	—	0	0	0	1	0	1	1	Read IS register.
16	OCW3 RR	0	—	0	0	0	1	0	1	0	Read requests register.
17	OCW3 SM	0	—	1	1	0	1	0	0	0	Set special mask mode.
18	OCW3 RSM	0	—	1	0	0	1	0	0	0	Reset special mask mode.

## Notes:

1. In the master mode  $\overline{SP}$  pin = 1, in slave mode  $\overline{SP}$  = 0.
2. (—) = do not care.

## Absolute Maximum Ratings

Ambient Temperature Under Bias ..... 0° C to 70° C  
 Storage Temperature ..... -65° C to +150° C  
 Voltage On Any Pin  
     With Respect to Ground ..... -0.5 V to +7 V  
 Power Dissipation ..... 1 Watt

### \*COMMENT:

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

## D.C. Characteristics: ( $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ ; $V_{CC} = 5\text{V} \pm 5\%$ )

SYMBOL	PARAMETER	MIN.	MAX.	UNITS	TEST CONDITIONS
$V_{IL}$	Input Low Voltage	-0.5	.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC} + .5\text{V}$	V	
$V_{OL}$	Output Low Voltage		.45	V	$I_{OL} = 2\text{ mA}$
$V_{OH}$	Output High Voltage	2.4		V	$I_{OH} = -400\text{ }\mu\text{A}$
$V_{OH-INT}$	Interrupt Output High Voltage	2.4		V	$I_{OH} = -400\text{ }\mu\text{A}$
		3.5		V	$I_{OH} = -50\text{ }\mu\text{A}$
$I_{IL}(IR_{0-7})$	Input Leakage Current for $IR_{0-7}$		-300	$\mu\text{A}$	$V_{IN} = 0\text{V}$
			10	$\mu\text{A}$	$V_{IN} = V_{CC}$
$I_{IL}$	Input Leakage Current for Other Inputs		10	$\mu\text{A}$	$V_{IN} = V_{CC}$ to 0V
$I_{LOL}$	Output Leakage Current		-10	$\mu\text{A}$	$V_{OUT} = 0.45\text{V}$
$I_{LOH}$	Output Leakage Current		10	$\mu\text{A}$	$V_{OUT} = V_{CC}$
$I_{CC}$	$V_{CC}$ Supply Current		85	mA	

## Capacitance $T_A = 25^\circ\text{C}$ ; $V_{CC} = \text{GND} = 0\text{V}$

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT	TEST CONDITIONS
$C_{IN}$	Input Capacitance			10	pF	$f_c = 1\text{ MHz}$
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to $V_{SS}$

**A.C. Characteristics:** ( $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ;  $V_{CC} = +5\text{V} \pm 5\%$ ,  $\text{GND} = 0\text{V}$ )

## BUS PARAMETERS

### READ

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AR}$	$\overline{\text{CS}}/\text{A}_0$ Stable before $\overline{\text{RD}}$ or $\overline{\text{INTA}}$	0		ns	
$t_{RA}$	$\overline{\text{CS}}/\text{A}_0$ Stable after $\overline{\text{RD}}$ or $\overline{\text{INTA}}$	0		ns	
$t_{RR}$	$\overline{\text{RD}}$ Pulse Width	300		ns	
$t_{RD}$	Data Valid from $\overline{\text{RD}}/\overline{\text{INTA}}$		300	ns	$\text{CL} = 100\text{ pF}$
$t_{DF}$	Data Float after $\overline{\text{RD}}/\overline{\text{INTA}}$	20	120	ns	$\text{CL} = 100\text{ pF}$ $\text{CL} = 20\text{ pF}$

### WRITE

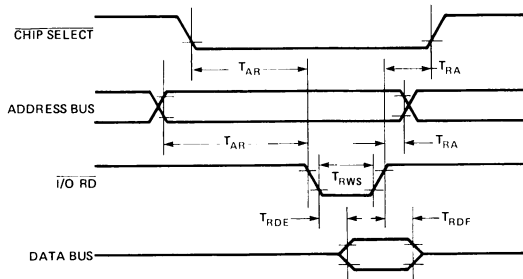
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{AW}$	$\text{A}_0$ Stable before $\overline{\text{WR}}$	0		ns	
$t_{WA}$	$\text{A}_0$ Stable after $\overline{\text{WR}}$	220		ns	
$t_{CW}$	$\overline{\text{CS}}$ Stable before $\overline{\text{WR}}$	0		ns	
$t_{WC}$	$\overline{\text{CS}}$ Stable after $\overline{\text{WR}}$	0		ns	
$t_{WW}$	$\overline{\text{WR}}$ Pulse Width	300		ns	
$t_{DW}$	Data Valid to $\overline{\text{WR}}$ (T.E.)	200		ns	
$t_{WD}$	Data Valid after $\overline{\text{WR}}$	-20		ns	

### OTHER TIMINGS

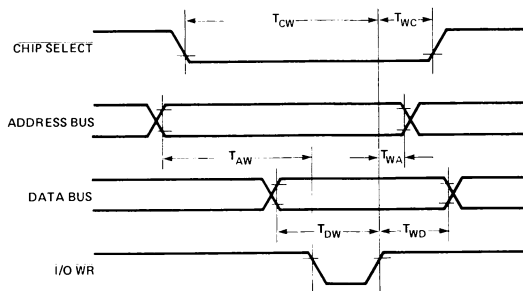
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{IW}$	Width of Interrupt Request Pulse	130		ns	
$t_{INT}$	$\text{INT} \uparrow$ after $\text{IR} \uparrow$	1.1		$\mu\text{s}$	
$t_{IC}$	Cascade Line Stable after $\overline{\text{INTA}} \uparrow$	500		ns	

## Waveforms

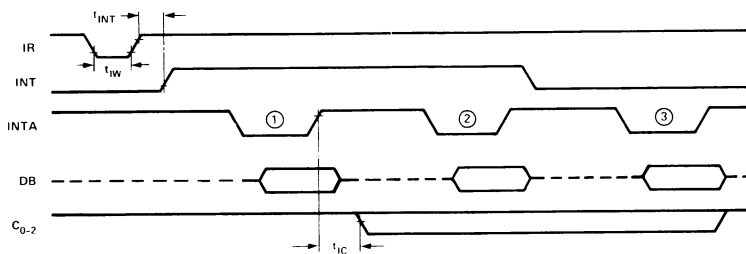
## READ TIMING



## WRITE TIMING



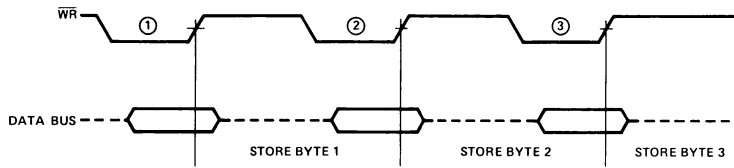
## OTHER TIMING



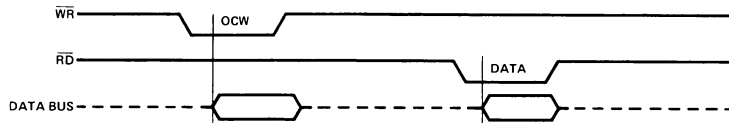
Note: Interrupt acknowledge  $\overline{INTA}$  sequence must remain "HIGH" (at least) until leading edge of first  $\overline{INTA}$ .

**PRELIMINARY**  
This document is a preliminary document. Some  
information may be subject to change.

## INITIALIZATION SEQUENCE



## READ STATUS/POLL MODE



# PROGRAMMABLE KEYBOARD/DISPLAY INTERFACE

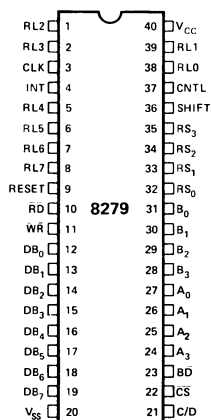
- Simultaneous Keyboard Display Operations
- Scanned Keyboard Mode
- Scanned Sensor Mode
- Strobed Input Entry Mode
- 8 Character Keyboard FIFO
- 2 Key or N Key Rollover with Contact Debounce
- Dual 8 or 16 Numerical Display
- Single 16 Character Display
- Right or Left Entry 16 Byte Display RAM
- Mode Programmable from CPU
- Programmable Scan Timing
- Interrupt Output on Key Entry

## Description

The 8279 is a general purpose programmable keyboard and display I/O interface device designed for use with the 8008, 8080 and 8048/8748 microprocessors. The keyboard portion can provide a scanned interface to a 64 contact key matrix which can be expanded to 128. The keyboard portion will also interface to an array of sensors or a strobed interface keyboard, such as the Hall effect and Ferrite variety. Key depressions can be 2 key or N key rollover. Keyboard entries are debounced and stored in an 8 character FIFO. If more than 8 characters are entered, over run status is set. Key entries set the interrupt output line to the CPU.

The display portion provides a scanned display interface for LED, incandescent and other popular display technologies. Both numeric and alphanumeric segment displays may be used as well as simple indicators. The 8279 has a 16 x 8 display RAM which can be organized into a dual 16 x 4. The RAM can be loaded or interrogated by the CPU. Both right entry, calculator and left entry typewriter display formats are possible. Both read and write of the display RAM can be done with auto-increment of the display RAM address.

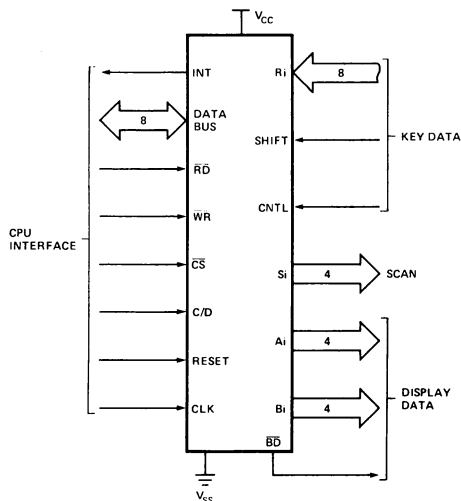
## PIN CONFIGURATION



## PIN NAMES

DB <sub>0-7</sub>	DATA BUS (BI-DIRECTIONAL)
CLK	CLOCK INPUT
RESET	RESET INPUT
CS	CHIP SELECT
RD	READ INPUT
WR	WRITE INPUT
C/D	COMMAND/DATA INPUT
INT	INTERRUPT OUTPUT
S <sub>0-3</sub>	SCAN OUTPUTS
R <sub>0-7</sub>	RETURN INPUTS
SHIFT	SHIFT INPUT
CNTL/STB	CONTROL/STROBE INPUT
A <sub>0-3</sub>	DISPLAY (A) OUTPUTS
B <sub>0-3</sub>	DISPLAY (B) OUTPUTS
BD	BLANK DISPLAY OUTPUT

## LOGIC SYMBOL



## 8279 BASIC FUNCTIONAL DESCRIPTION

### Introduction

Since data input and display are an integral part of many microprocessor designs, the system designer needs an interface that can control these functions without placing a large load on the CPU. The 8279 provides this function for 8-bit microprocessors such as the 8080.

The 8279 has two sections: keyboard and display. The keyboard section can interface to regular typewriter style keyboards or random toggle or thumb switches. The display section drives alphanumeric displays or a bank of indicator lights. Thus the CPU is relieved from scanning the keyboard or refreshing the display.

The 8279 is designed to directly connect to the 8080 bus. The CPU can program all operating modes for the 8279. These modes include:

### Input Modes

- Scanned Keyboard — with encoded (8 x 8 x 4 key keyboard) or decoded (4 x 8 x 4 key keyboard) scan lines. A key depression generates a 6-bit encoding of key position. Position and shift and control status are stored in the FIFO. Keys are automatically debounced with 2-key or N-key rollover.

- Scanned Sensor Matrix — with encoded (8 x 8 matrix switches) or decoded (4 x 8 matrix switches) scan lines. Key status (open or closed) stored in RAM addressable by CPU.
- Strobed Input — Data on return lines during control line strobe is transferred to FIFO.

### Output Modes

- 8 or 16 character multiplexed displays that can be organized as dual 4-bit or single 8-bit.
- Right entry or left entry display formats.

Other features of the 8279 include:

- Mode programming from the CPU.
- Programmable clock to match the 8279 scan times to the CPU cycle time.
- Interrupt output to signal CPU when there is keyboard or sensor data available.
- An 8 byte FIFO to store keyboard information.
- 16 byte internal Display RAM for display refresh. This RAM can also be read by the CPU.

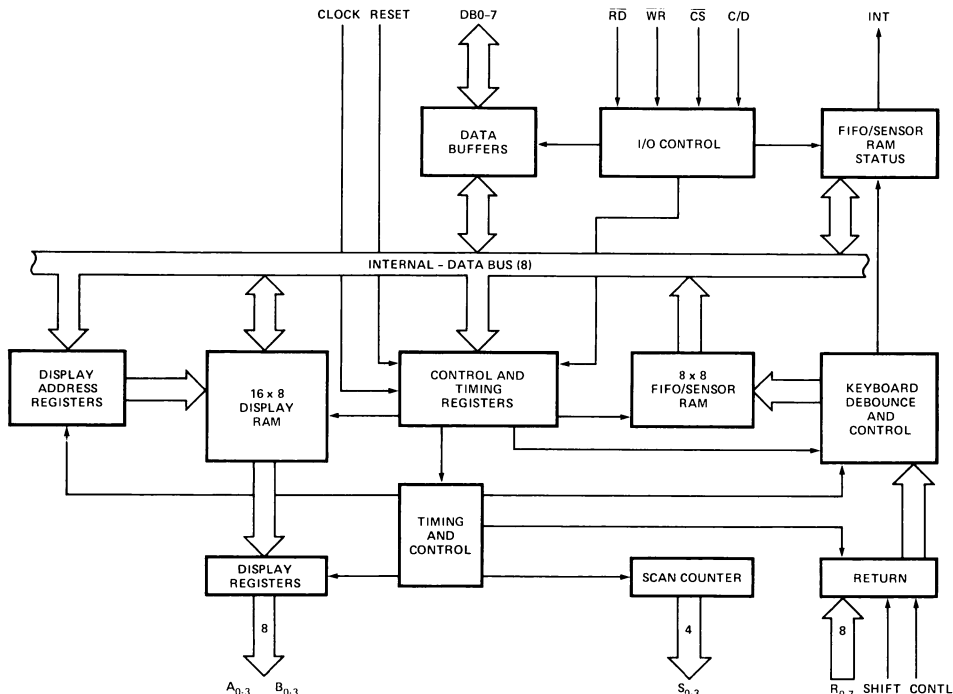


FIGURE 1. 8279 BLOCK DIAGRAM



## Hardware Description

The 8279 is packaged in a 40 pin DIP. The following is a functional description of each pin.

No. Of Pins	Designation	Function
8	DB0-DB7	Bi-directional data bus. All data and commands between the CPU and the 8279 are transmitted on these lines.
1	CLK	Clock from system used to generate internal timing.
1	RESET	A high signal on this pin resets the 8279.
1	$\overline{CS}$	Chip Select. A low on this pin enables the interface functions to receive or transmit.
1	C/D	Command/Data. A high on this line indicates the signals in or out are interpreted as a command. A low indicates that they are data.
2	$\overline{RD}$ , $\overline{WR}$	Input/Output read and write. These signals enable the data buffers to either send data to the external bus or receive it from the external bus.
1	INT	Interrupt Output. In a keyboard mode, the interrupt line is high when there is data in the FIFO/ Sensor RAM. The interrupt line goes low with each FIFO/ Sensor RAM read and returns high if there is still information in the RAM. In a sensor mode, the interrupt line goes high whenever a change in a sensor is detected.
2	$V_{SS}$ , $V_{CC}$	Ground and +5 $\pm 10\%$ power supply pins.
4	S0-S3	Scan outputs which are used to scan the key switch or sensor matrix and the display digits. These lines can be either encoded (1 of 16) or decoded (1 of 4).
8	R0-R7	Return line inputs which are connected to the scan lines through the keys or sensor switches. They have active internal pullups to keep them high until a switch closure pulls one low. They also serve as an 8-bit input in the Strobed Input mode.
1	SHIFT	The shift input status is stored along with the key position on key closure in the Scanned Keyboard modes.

No. Of Pins	Designation	Function
1	CNTL/STB	For keyboard modes this line is used as a control input and stored like status on a key closure. The line is also the strobe line that enters the data into the FIFO in the Strobed Input mode.
4	A0-A3	These two ports are the outputs for the 16 x 4 display refresh registers. The data from these outputs is synchronized to the scan lines (S0-S3) for multiplexed digit displays. The two 4 bit ports may be blanked independently. These two ports may also be considered as one 8 bit port.
4	B0-B3	
1	$\overline{BD}$	Blank Display. This output is used to blank the display during digit switching or by a display blanking command.

## Principles of Operation

The following is a description of the major elements of the 8279 Programmable Keyboard/Display interface device. Refer to the block diagram in Figure 1.

### I/O Control and Data Buffers

The I/O control section uses the  $\overline{CS}$ , C/D,  $\overline{RD}$  and  $\overline{WR}$  lines to control data flow to and from the various internal registers and buffers. All data flow to and from the 8279 is enabled by  $\overline{CS}$ . The character of the information, given or desired by the CPU, is identified by C/D. A logic one means the information is a command or status. A logic zero means the information is data.  $\overline{RD}$  and  $\overline{WR}$  determine the direction of data flow through the Data Buffers. The Data Buffers are bi-directional buffers that connect the internal bus to the external bus. When the chip is not selected ( $\overline{CS} = 1$ ), the devices are in a high impedance state. The drivers input during  $\overline{WR} \bullet \overline{CS}$  and output during  $\overline{RD} \bullet \overline{CS}$ .

### Control and Timing Registers and Timing Control

These registers store the keyboard and display modes and other operating conditions programmed by the CPU. The modes are programmed by presenting the proper command on the data lines with C/D = 1 and then sending a  $\overline{WR}$ . The command is latched on the rising edge of  $\overline{WR}$ . The command is then decoded and the appropriate function is set. The timing control contains the basic timing counter chain. The first counter is  $a \div N$  prescaler that can be programmed to match the CPU cycle time to the internal timing. The prescaler is software programmed to a value between 2 and 31. A value which yields an internal frequency of 100 kHz gives a 5.1 ms keyboard scan time and a 10.3 ms debounce time. The other counters divide down the basic internal frequency to provide the proper key scan, row scan, keyboard matrix scan, and display scan times.

## Scan Counter

The scan counter has two modes. In the encoded mode, the counter provides a binary count that must be externally decoded to provide the scan lines for the keyboard and display. In the decoded mode, the scan counter decodes the least significant 2 bits and provides a decoded 1 of 4 scan. Note that when the keyboard is in decoded scan so is the display. This means that only the first 4 characters in the Display RAM are displayed.

## Return Buffers and Keyboard Debounce and Control

The 8 return lines are buffered and latched by the Return Buffers. In the keyboard mode, these lines are scanned, looking for key closures in that row. If the debounce circuit detects a closed switch, it waits about 10 msec to check if the switch remains closed. If it does, the address of the switch in the matrix plus the status of SHIFT and CONTROL are transferred to the FIFO. In the scanned Sensor Matrix modes, the contents of the return lines is directly transferred to the corresponding row of the Sensor RAM (FIFO) each key scan time. In Strobed Input mode, the contents of the return lines are transferred to the FIFO on the rising edge of the CNTL/STB line pulse.

## FIFO/Sensor RAM and Status

This block is a dual function 8 x 8 RAM. In Keyboard or Strobed Input modes, it is a FIFO. Each new entry is written into successive RAM positions and each is then read in order of entry. FIFO status keeps track of the number of characters in the FIFO and whether it is full or empty. Too many reads or writes will be recognized as an error. The status can be read by an RD with  $\overline{CS}$  low and C/D high. The status logic also provides an INT signal when the FIFO is not empty. In Scanned Sensor Matrix mode, the memory is a Sensor RAM. Each row of the Sensor RAM is loaded with the status of the corresponding row of sensor in the sensor matrix. In this mode, INT is high if a change in a sensor is detected.

## Display Address Registers and Display RAM

The Display Address Registers hold the address of the word currently being written or read by the CPU and the two 4-bit nibbles being displayed. The read/write addresses are programmed by CPU command. They also can be set to auto increment after each read or write. The Display RAM can be directly read by the CPU after the correct mode and address is set. The addresses for the A and B nibbles are automatically updated by the 8279 to match data entry by the CPU. The A and B nibbles can be entered independently or as one word, according to the mode that is set by the CPU. Data entry to the display can be set to either left or right entry. See Interface Considerations for details.

## Software Operation

### 8279 Commands

The following commands program the 8279 operating modes. The commands are sent on the Data Bus with  $\overline{CS}$  low and C/D high and are loaded to the 8279 on the rising edge of  $\overline{WR}$ .

### Keyboard/Display Mode Set

	MSB				LSB			
Code:	0	0	0	D	D	K	K	K

Where DD is the Display Mode and KKK is the Keyboard Mode.

### DD

- 0 0 8 8-bit character display — Left entry
- 0 1 16 8-bit character display — Left entry\*
- 1 0 8 8-bit character display — Right entry
- 1 1 16 8-bit character display — Right entry

For description of right and left entry, see Interface Considerations. Note that when decoded scan is set in keyboard mode, the display is reduced to 4 characters independent of display mode set.

### KKK

- 0 0 0 Encoded Scan Keyboard — 2 Key Rollover\*
- 0 0 1 Decoded Scan Keyboard — 2-Key Rollover
- 0 1 0 Encoded Scan Keyboard — N-Key Rollover
- 0 1 1 Decoded Scan Keyboard — N-Key Rollover
- 1 0 0 Encoded Scan Sensor Matrix
- 1 0 1 Decoded Scan Sensor Matrix
- 1 1 0 Strobed Input, Encoded Display Scan
- 1 1 1 Strobed Input, Decoded Display Scan

### Program Clock

Code:	0	0	1	P	P	P	P	P
-------	---	---	---	---	---	---	---	---

Where P P P P P is the prescaler value 2 to 31. The programmable prescaler divides the external clock by P P P P P to get the basic internal frequency. Choosing a divisor that yields 100 KHz will give the specified scan and debounce times. Default after a reset pulse (but not a program clear) is 31.

### Read FIFO/Sensor RAM

Code:	0	1	0	AI	X	A	A	A
-------	---	---	---	----	---	---	---	---

X = Don't Care

Where AI is the Auto-Increment flag for the Sensor RAM and AAA is the row that is going to be read by the CPU. AI and AAA are used only if the mode is set to Sensor Matrix. This command is used to specify that the source of data reads ( $\overline{CS} \bullet \overline{RD} \bullet \overline{CD}$ ) by the CPU is the FIFO/Sensor RAM. No additional commands are necessary as long as  
 \*Default after reset.

data is desired from the FIFO/Sensor RAM. Another command is necessary if reading is desired from a different row than has been selected. If AI is a one, the row select counter will be incremented after each read so the next read will be from the next Sensor RAM row.

In the Auto Increment mode for reading data from the FIFO/Sensor RAM, each read advances the address by one so that the next read is from the next character. This Auto Incrementing has no effect on the display.

#### Read Display RAM

Code:

0	1	1	AI	A	A	A	A
---	---	---	----	---	---	---	---

Where AI is the Auto-Increment flag for the Display RAM and AAAA is the character that the CPU is going to read next. Since the CPU uses the same counter for reading and writing, this command also sets the next write location and Auto-Increment mode. This command is used to specify the display RAM as the data source for CPU data reads. If AI is set, the character address will be incremented after each read (or write) so that the next read (or write) will be from (to) the next character.

#### Write Display RAM

Code:

1	0	0	AI	A	A	A	A
---	---	---	----	---	---	---	---

Where AI is the Auto-Increment flag for the Display RAM and AAAA is the character that the CPU is going to write next. The addressing and Auto-Increment are identical to Read Display RAM. The difference is that Write Display RAM does not affect the source of CPU reads. The CPU will read from whichever RAM (Display or FIFO/Sensor) was last specified. This command will, however, change the location the next Display RAM read will be from if that source was specified.

#### Display Write Inhibit/Blanking

Code:

1	0	1	X	IW	IW	BL	BL
				A	B	A	B

Where IW is Inhibit Writing (nibble A or B) and BL is Blanking (nibble A or B). If the display is being used as a dual 4-bit display, then it is necessary to mask one of the 4-bit halves so that entries to the Display from the CPU do not affect the other half. The IW flags allow the programmer to do this. It is also useful to be able to blank either half when that half is not to be displayed. The BL flags blank the display. The next command sets the output code to be used as a "blank". Default after reset is all zeros. Note that to blank a display formatted as a single 8-bit output, it is necessary to set both BL flags to entirely blank the display. A "1" sets the flag. Reissuing the command with a "0" resets the flag.

#### Clear

Code:

1	1	0	C <sub>D</sub>	C <sub>D</sub>	C <sub>D</sub>	C <sub>F</sub>	C <sub>A</sub>
---	---	---	----------------	----------------	----------------	----------------	----------------

Where C<sub>D</sub> is Clear Display, C<sub>F</sub> is Clear FIFO Status (including interrupt), and C<sub>A</sub> is Clear All. C<sub>D</sub> is used to

clear all positions of the Display RAM to a programmable code. All ones, all zeros and hexadecimal 20 are possible. The 2 least significant bits of C<sub>D</sub> are also used to specify the blanking code (see below).

C <sub>D</sub>	C <sub>D</sub>	C <sub>D</sub>	
0	X		All Zeros (X = Don't Care)
1	0		Hex 20 (0010 0000)
1	1		All Ones
↑			Enable clear display when = 1 (or by C <sub>A</sub> = 1)

Clearing the display takes one display scan. During this time the CPU cannot write to the Display RAM. The MSB of the FIFO status word will be set during this time.

C<sub>F</sub> sets the FIFO status to empty and resets the interrupt output line. After execution of a clear command with C<sub>F</sub> set, the Sensor Matrix mode RAM pointer will be set to row 0.

C<sub>A</sub> has the combined effect of C<sub>D</sub> and C<sub>F</sub>. C<sub>A</sub> uses the C<sub>D</sub> clearing code to determine how to clear the Display RAM. C<sub>A</sub> also resets the internal timing chain to resynchronize it.

#### End Interrupt/Error Mode Set

Code:

1	1	1	E	X	X	X	X
---	---	---	---	---	---	---	---

X = Don't care.

For the sensor matrix modes this command lowers the INT line and enables further writing into RAM. (The INT line would have been raised upon the detection of a change in a sensor value. This would have also inhibited further writing into the RAM until reset.)

For the N-key rollover mode — if the E bit is programmed to "1" the chip will operate in the special Error mode. (For further details, see Interface Considerations Section.)

#### Status Word

The status word contains the FIFO status, error, and display unavailable signals. This word is read by the CPU when C/D is high and CS and RD are low. See Interface Considerations for more detail on status word.

#### Data Read

Data is read when C/D, CS and RD are all low. The source of the data is specified by the Read FIFO or Read Display commands. The trailing edge of RD will cause the address of the RAM being read to be incremented if the Auto-Increment flag is set. FIFO reads always increment (if no error occurs) independent of AI.

#### Data Write

Data that is written with C/D, CS and WR low is always written to the Display RAM. The address is specified by the latest Read Display or Write Display command. Auto-Incrementing on the rising edge of WR occurs if AI set by the latest display command.

## INTERFACE CONSIDERATIONS

### A. Scanned Keyboard Mode, 2-Key Rollover

There are three possible combinations of conditions that can occur during debounce scanning. When a key is depressed, the debounce logic is set. A full scan of the keyboard is ignored, then other depressed keys are looked for. If none are encountered, it is a single key depression and the key position is entered into the FIFO along with the status of CNTL and SHIFT lines. If the FIFO was empty, INT will be set to signal the CPU that there is an entry in the FIFO. If the FIFO was full, the key will not be entered and the error flag will be set. If another closed switch is encountered, no entry to the FIFO can occur. If all other keys are released before this one, then it will be entered to the FIFO. If this key is released before any other, it will be entirely ignored. A key is entered to the FIFO only once per depression, no matter how many keys were pressed along with it or in what order they were released. If two keys are depressed within the debounce cycle, it is a simultaneous depression. Neither key will be recognized until one key remains depressed alone. The last key will be treated as a single key depression.

### B. Scanned Keyboard Mode, N-Key Rollover

With N-key Rollover each key depression is treated independently from all others. When a key is depressed, the debounce circuit waits 2 keyboard scans and then checks to see if the key is still down. If it is, the key is entered into the FIFO. Any number of keys can be depressed and another can be recognized and entered into the FIFO. If a simultaneous depression occurs, the keys are recognized and entered according to the order the keyboard scan found them.

### C. Scanned Keyboard — Special Error Modes

For N-key rollover mode the user can program a special error mode. This is done by the "End Interrupt/Error Mode Set" command. The debounce cycle and key-validity check are as in normal N-key mode. If during a single debounce cycle, two keys are found depressed, this is considered a simultaneous multiple depression, and sets an error flag. This flag will prevent any further writing into the FIFO and will set interrupt (if not yet set). The error flag could be read in this mode by reading the FIFO STATUS word. (See "FIFO STATUS" for further details.) The error flag is reset by sending the normal CLEAR command with CF = 1.

### D. Sensor Matrix Mode

In Sensor Matrix mode, the debounce logic is inhibited. The status of the sensor switch is inputted directly to the Sensor RAM. In this way the Sensor RAM keeps an image of the state of the switches in the sensor matrix. Although debouncing is not provided, this mode has the advantage that the CPU knows how long the sensor was closed and when it was released. A keyboard mode can only indicate a validated closure. To make the software easier, the designer should functionally group the sensors by row since this is the format in which the CPU will read them. The INT line goes high if any sensor value change is detected at the end of a sensor matrix scan. The INT line is cleared by the first Data Read Command if the Auto-

Increment flag is set to zero, or by the End Interrupt command if the Auto-Increment flag is set to one.

### E. Data Format

In the Scanned Keyboard mode, the character entered into the FIFO corresponds to the position of the switch in the keyboard plus the status of the CNTL and SHIFT lines. CNTL is the MSB of the character and SHIFT is the next most significant bit. The next three bits are from the scan counter and indicate the row the key was found in. The last three bits are from the column counter and indicate to which return line the key was connected.

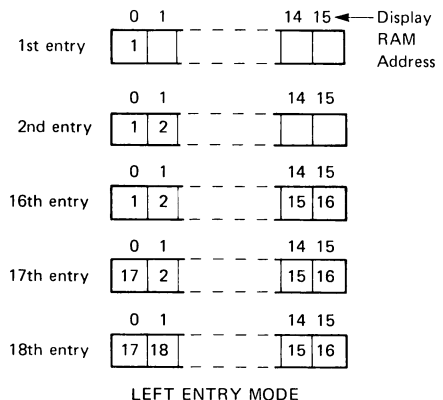
In Sensor Matrix mode, the data on the return lines is entered directly in the row of the Sensor RAM that corresponds to the row in the matrix being scanned. Therefore, each switch position maps directly to a Sensor RAM position. The SHIFT and CNTL inputs are ignored in this mode. Note that switches are not necessarily the only thing that can be connected to the return lines in this mode. Any logic that can be triggered by the scan lines can enter data to the return line inputs. Eight multiplexed input ports could be tied to the return lines and scanned by the 8279.

In Strobed Input mode, the data is also entered to the FIFO from the return lines. The data is entered by the rising edge of a CNTL/STB line pulse. Data can come from another encoded keyboard or simple switch matrix. The return lines can also be used as a general purpose strobed input.

### F. Display

#### Left Entry

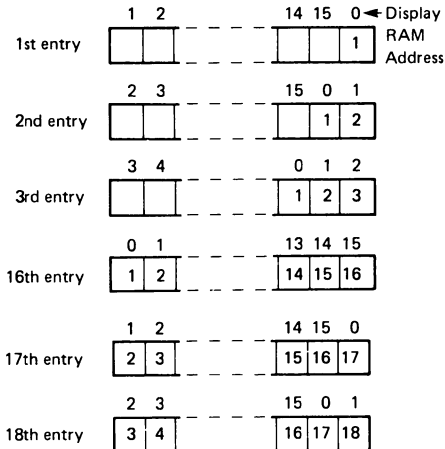
Left Entry mode is the simplest display format in that each display position directly corresponds to a byte (or nibble) in the Display RAM. Address 0 in the RAM is the left-most display character and address 15 (or address 7 in 8 character display) is the right most display character. Entering characters from position zero causes the display to fill from the left. The 17th (9th) character is entered back in the left most position and filling again proceeds from there.



**PRELIMINARY**  
 Note: This document is preliminary and subject to change. Some parameters may be subject to change.

## Right Entry

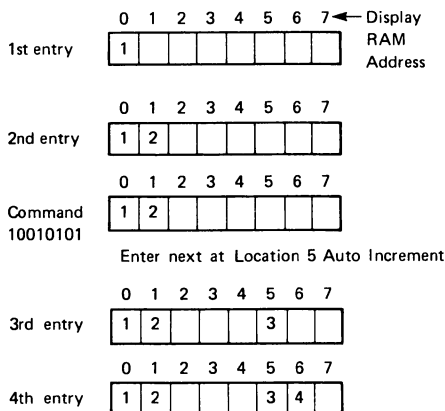
Right entry is the method used by most electronic calculators. The first entry is placed in the right most display character. The next entry is also placed in the right most character after the display is shifted left one character. The left most character is shifted off the end and is lost.



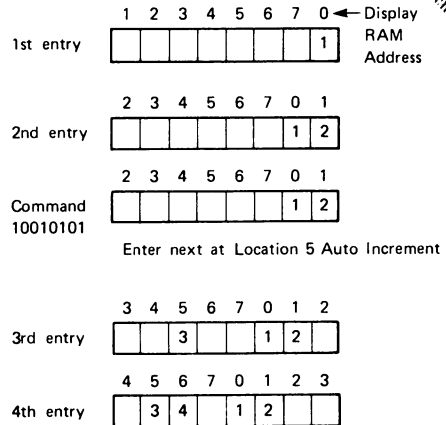
Note that now the display position and register address do not correspond. Consequently, entering a character to an arbitrary position in the Auto Increment mode may have unexpected results. Entry starting at Display RAM address 0 with sequential entry is recommended.

## Auto Increment

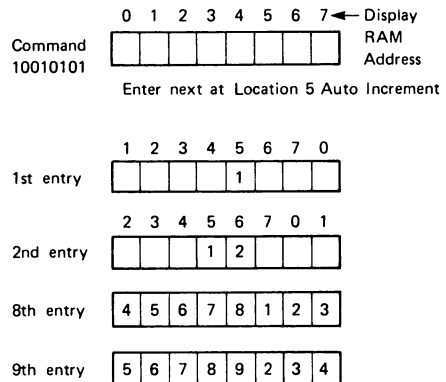
In the Left Entry mode, Auto Incrementing causes the address where the CPU will next write to be incremented by one and the character appears in the next location. With non-Auto Incrementing the entry is both to the same RAM address and display position. Entry to an arbitrary address in the Auto Increment mode has no undesirable side effects and the result is predictable:



In the Right Entry mode, Auto Incrementing and non Incrementing have the same effect as in the Left Entry except if the address sequence is interrupted:



Starting at an arbitrary location operates as shown below:



Entry appears to be from the initial entry point.

## 8/16 Character Display Formats

If the display mode is set to an 8 character display, the on duty-cycle is double what it would be for a 16 character display (e.g., 5.1 ms scan time for 8 characters vs. 10.3 ms for 16 characters with 100 kHz internal frequency).

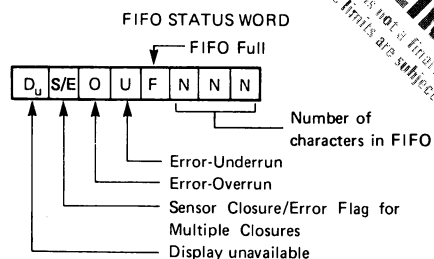
## G. FIFO Status

FIFO status is used in the Keyboard and Strobed Input modes to indicate the number of characters in the FIFO and to indicate whether an error has occurred. There are two types of errors possible: overrun and underrun. Overrun occurs when the entry of another character into a full FIFO is attempted. Underrun occurs when the CPU tries to read an empty FIFO.

The FIFO status word also has a bit to indicate that the Display RAM was unavailable because a Clear Display or Clear All command had not completed its clearing operation.

In a Sensor Matrix mode, a bit is set in the FIFO status word to indicate that at least one sensor closure indication is contained in the Sensor RAM.

In Special Error Mode the S/E bit is showing the error flag and serves as an indication to whether a simultaneous multiple closure error has occurred.



## APPLICATIONS

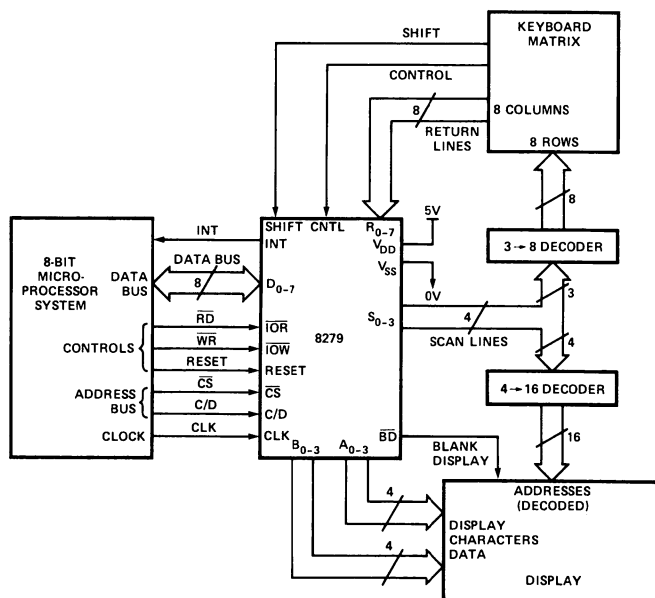


FIGURE 2. GENERAL BLOCK DIAGRAM

## ABSOLUTE MAXIMUM RATINGS\*

Ambient Temperature . . . . . 0°C to 70°C  
 Storage Temperature . . . . . -65°C to 125°C  
 Voltage on any Pin with  
   Respect to Ground . . . . . -0.5V to +7V  
 Power Dissipation . . . . . 1 Watt

### \*COMMENT

*Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

## D.C. and OPERATING CHARACTERISTICS

$T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$ ,  $V_{CC} = +5\text{V} \pm 10\%$ ,  $V_{SS} = 0\text{V}$

SYMBOL	PARAMETER	LIMITS			UNIT	TEST CONDITIONS
		MIN.	TYP.	MAX.		
$V_{OL}$	Output Low Voltage			0.45	V	$I_{OL} = 2.2\text{ mA}$
$V_{OH}$	Output High Voltage	2.4			V	$I_{OH} = -400\text{ }\mu\text{A}$
$V_{ILV}$	Input Low Voltage (for all inputs but R's)	$V_{SS} - 0.5$		0.8	V	
$V_{IL2}$	Input Low Voltage for Return Lines	$V_{SS} - 0.5$		1.4	V	
$V_{IH}$	Input High Voltage	2.0			V	
$I_{ILa}$	Input Leakage Current			$\pm 10$	$\mu\text{A}$	$V_{in} = V_{CC}$
$I_{FL}$	Output Float Leakage			$\pm 10$	$\mu\text{A}$	$V_{in} = V_{CC}$ or $V_{in} = V_{SS} + .45\text{ V}$
$I_{CC}$	Power Supply Current			120	mA	
$I_{ILL}$	Input Leakage Current on Return Lines, Shifts and Control			+10 -100	$\mu\text{A}$ $\mu\text{A}$	$V_{in} = V_{CC}$ $V_{in} = V_{SS}$
$V_{OHL}$	Output High Voltage on Interrupt Line	3.5			V	$I_{OH} = -100\text{ }\mu\text{A}$

## A.C. CHARACTERISTICS

$T_A = 0^\circ$  to  $70^\circ\text{C}$ ,  $V_{CC} = \pm 10\%$ ,  $V_{SS} = 0\text{V}$

SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
$t_{RCY}$	Read Cycle Time	1000		nsec	
$t_{RD}$	$\overline{IOR}$ to Data Out Stable		150	nsec	100 pF on Data Bus
$t_{CD}$	$\overline{CS}$ to Data Out Stable		250	nsec	100 pF on Data Bus
$t_{CR}$	C/D to $\overline{IOR}$ Set Up Time	0		nsec	
$t_{RC}$	C/D to $\overline{IOR}$ Hold Time	0		nsec	
$t_{DW}$	Data Set Up to $\overline{IOW}$ Trailing Edge	150		nsec	
$t_{CW}$	C/D Set Up to $\overline{IOW}$	0		nsec	
$t_{WW}$	$\overline{IOW}$ Pulse Width	250		nsec	
$t_{WC}$	C/D Hold from $\overline{IOW}$	0		nsec	
$t_{WD}$	Data Hold from $\overline{IOW}$	-20		nsec	
$t_{\phi W}$	Clock Pulse Width	120		nsec	
$t_{CY}$	Clock Period	320		nsec	
$t_{CSR}$	$\overline{CS}$ Stable before $\overline{IOR}$	0		nsec	
$t_{RCS}$	$\overline{CS}$ Hold after $\overline{IOR}$	0		nsec	
$t_{RR}$	$\overline{IOR}$ Width	300		nsec	
$t_{CDD}$	C/D to Data Output Stable		250	nsec	$C_L = 100\text{ pF}$
$t_{RDF}$	Data Float after $\overline{IOR}$		100	nsec	$C_L = 100\text{ pF}$
		10		nsec	$C_L = 15\text{ pF}$
$t_{CSW}$	$\overline{CS}$ Stable before $\overline{IOW}$	0		nsec	
$t_{WCS}$	$\overline{CS}$ Hold from $\overline{IOW}$	0		nsec	

## CAPACITANCE

SYMBOL	TEST	TYP.	MAX.	UNIT	TEST CONDITIONS
$C_{in}$	Input Capacitance	5	10	pF	$V_{in} = V_{CC}$
$C_{out}$	Output Capacitance	10	20	pF	$V_{out} = V_{CC}$

## A.C. TEST CONDITIONS

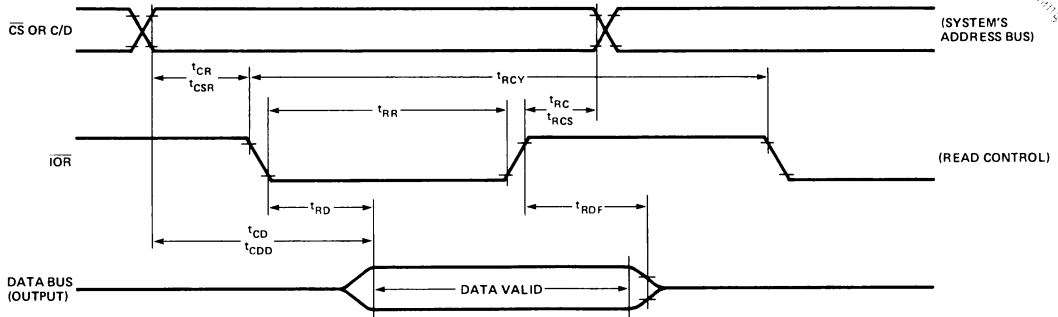
Output Load . . . . . 1 TTL Gate, and  $C_{LOAD} = 100\text{ pF}$   
 Input Pulse Levels . . . . . 0.8 to 2.0V  
 Input Pulse Rise and Fall Times . . . . . (10% to 90%) 20 nS  
 Timing Measurement Reference Level  
   Input . . . . . 1.5V  
   Output . . . . . 0.45V to 2.2V

Keyboard Scan Time: 5.1 msec  
 Keyboard Debounce Time: 10.3 msec  
 Key Scan Time: 80  $\mu\text{sec}$   
 Display Scan Time: 10.3 msec  
 Digit-on Time: 480  $\mu\text{sec}$   
 Blanking Time: 160  $\mu\text{sec}$   
 Internal Clock Cycle: 10  $\mu\text{sec}$

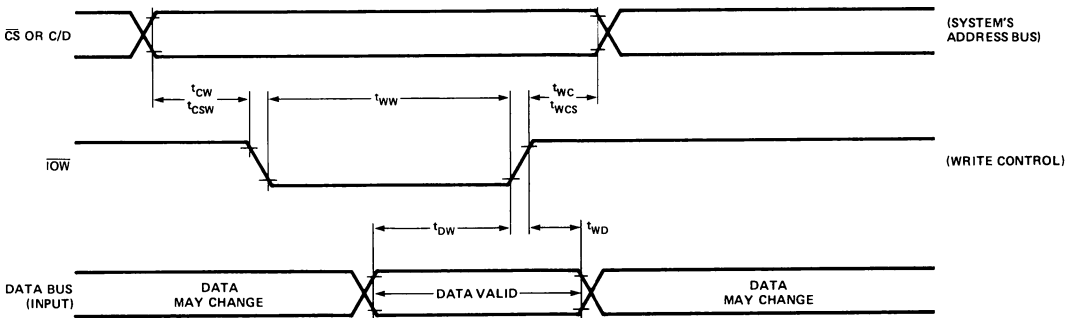


## WAVEFORMS

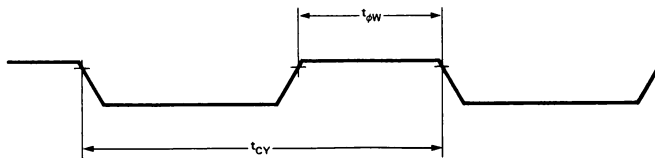
### 1. Read Operation

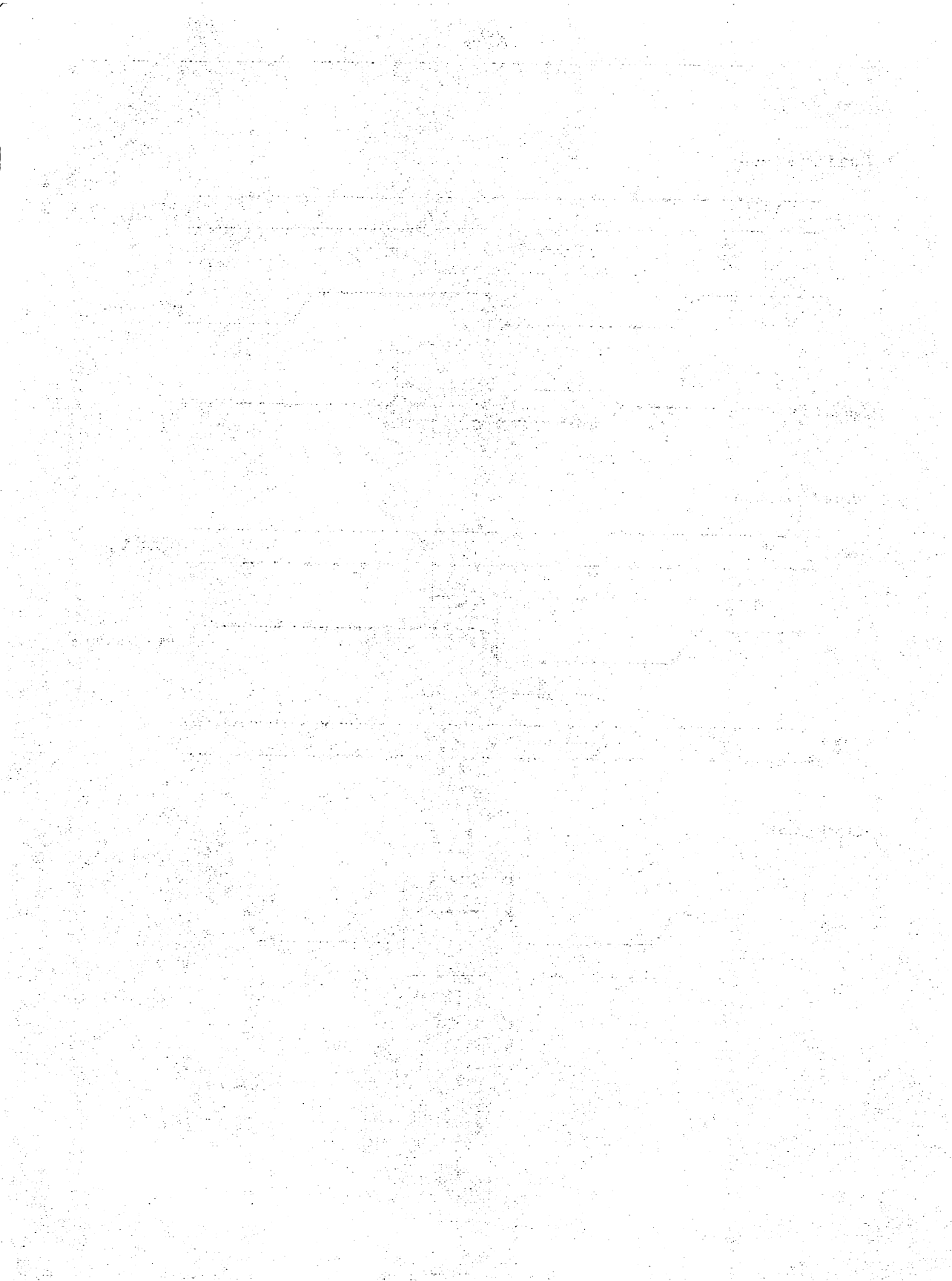


### 2. Write Operation



### 3. Clock Input





**Appendices**  
**PACKAGING INFORMATION**  
**AND**  
**ORDERING INFORMATION**



## APPENDICES

Packaging Information .....	A1-1
Ordering Information .....	A2-1

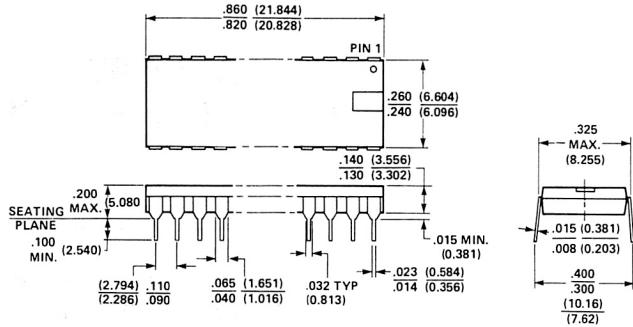
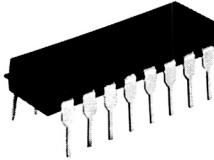
# APPENDIX 1

## PACKAGING INFORMATION

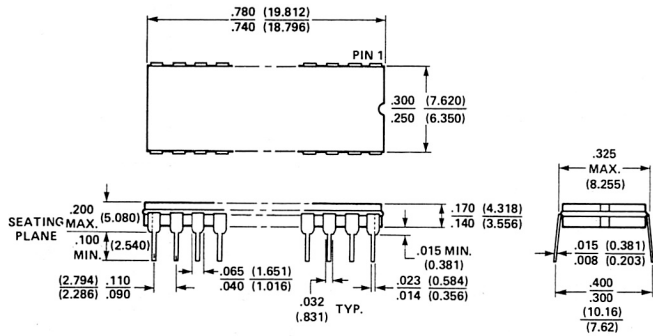
	Intel Product Number	Standard Package Type	Number of Pins	Comments
Microcomputers	8048 8748 8035	D P B D P	40 40 40	Available 2Q 1977
Memory and I/O Expanders	8355 8755 8155	D P C D P	40 40 40	
I/O Expander	8243	D P	24	
Standard ROMs	8308 8316A	D P C D P	24 24	
Standard EPROM	8708	B	24	
Standard RAMs	8111A-4 8101A-4 5101	C D P B C P B P	18 22 24	
Standard I/O	8212 8255 8251	D P C P C D P	24 40 28	
Standard Peripherals	8205 8214 8216 8226 8253 8259 8279	D P D P D P D P C D P D	16 24 16 16 24 28 40	

B = Black Ceramic   C = Ceramic   D = Ceramic DIP   P = Plastic

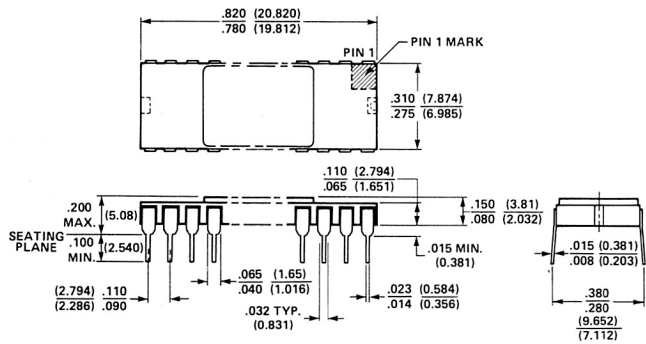
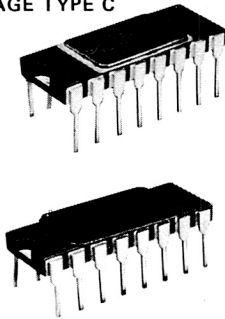
**16-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**



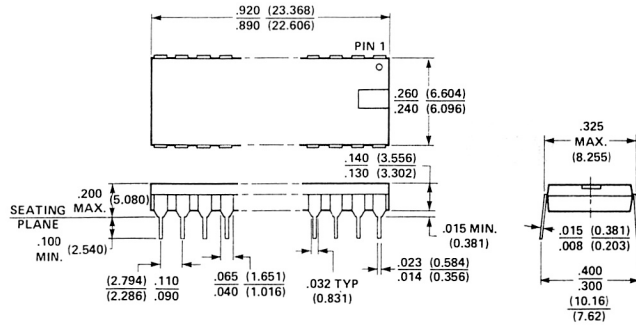
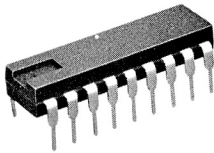
**16-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE D**



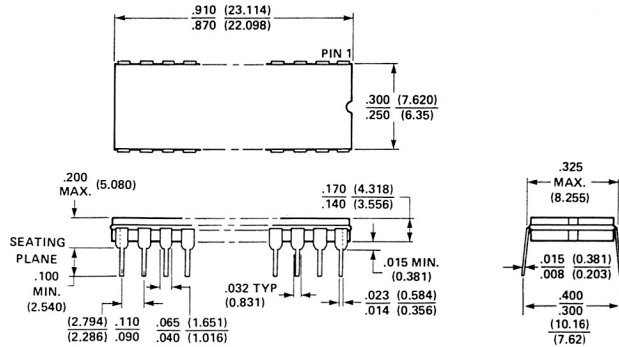
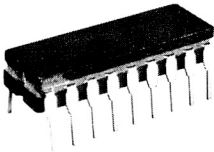
**16-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C**



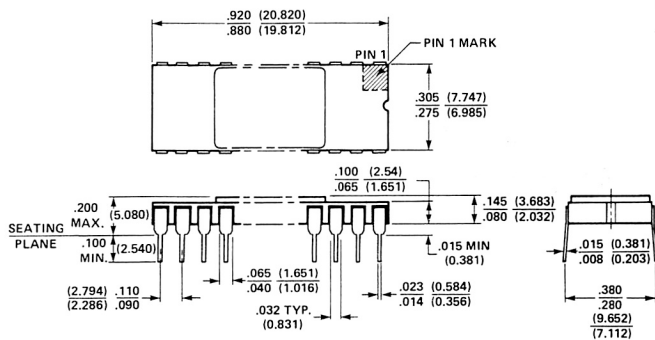
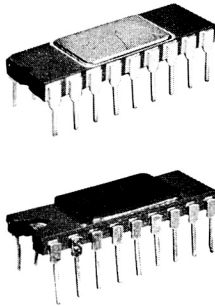
**18-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**



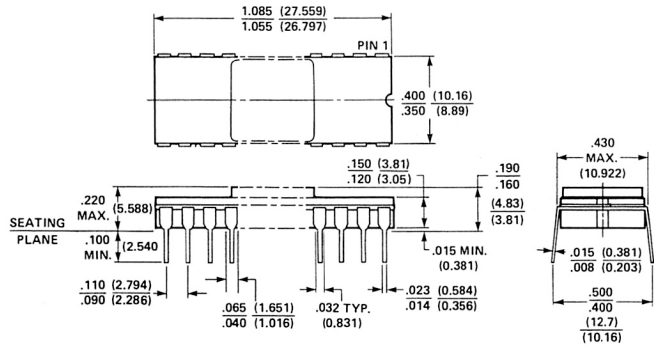
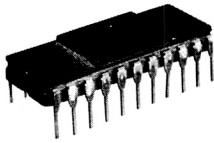
**18-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE D**



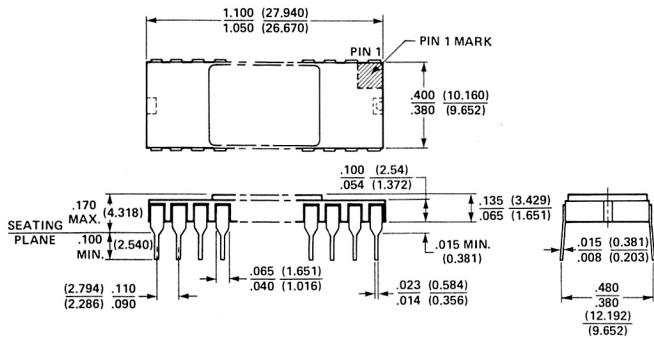
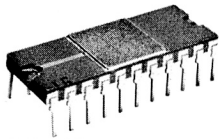
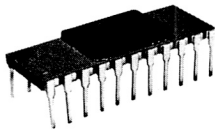
**18-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C**



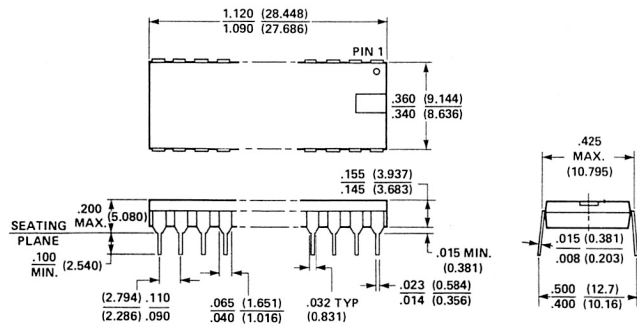
**22-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE B**



**22-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C**

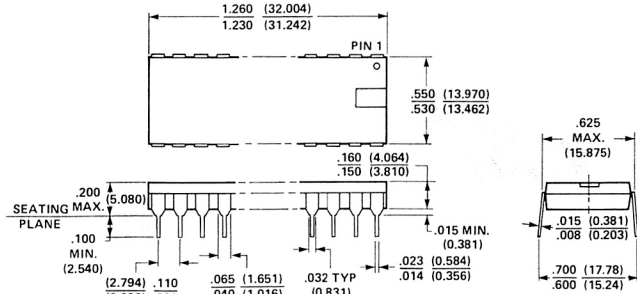
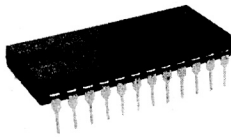


**22-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**

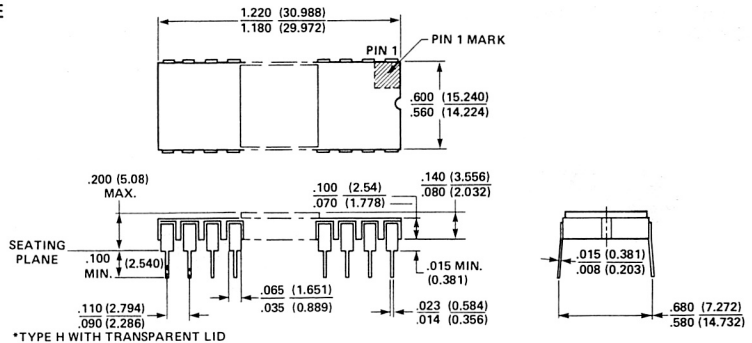
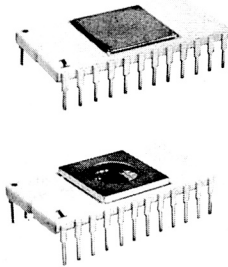




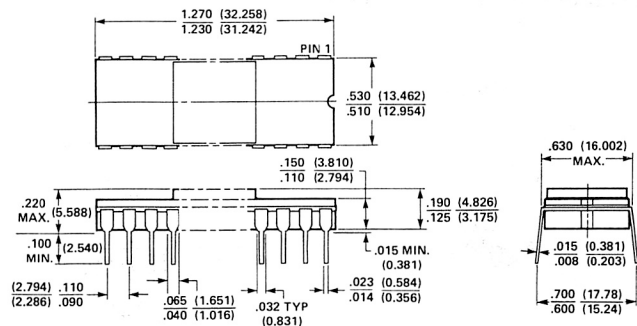
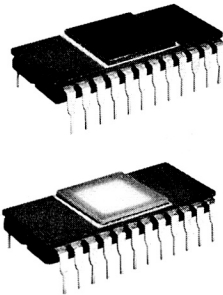
**24-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**



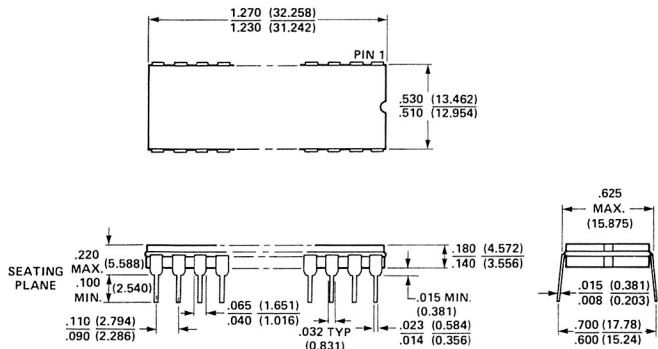
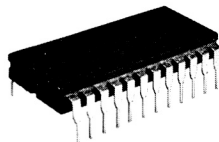
**24-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C OR H\***



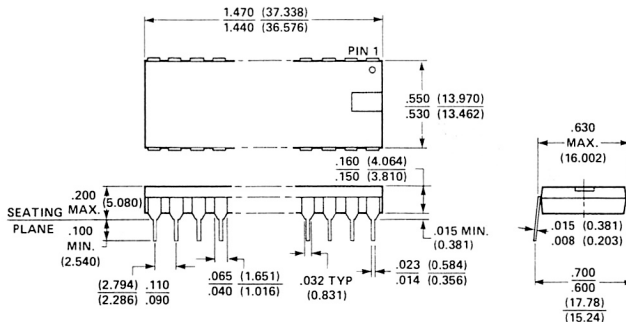
**24-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE B**



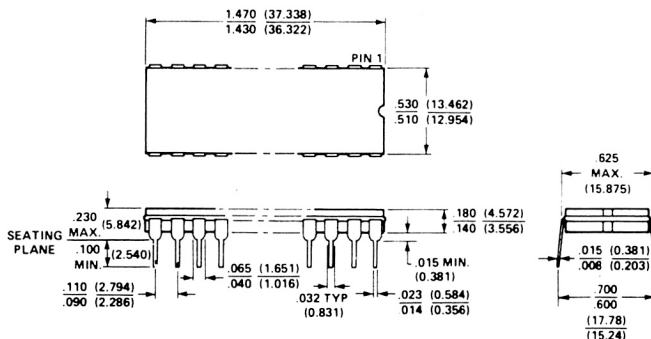
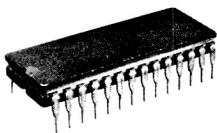
**24-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE D**



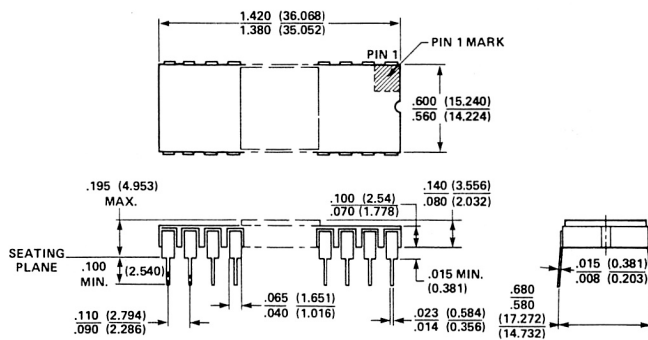
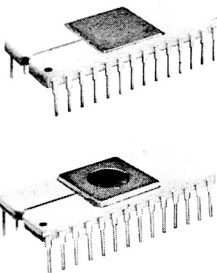
**28-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**



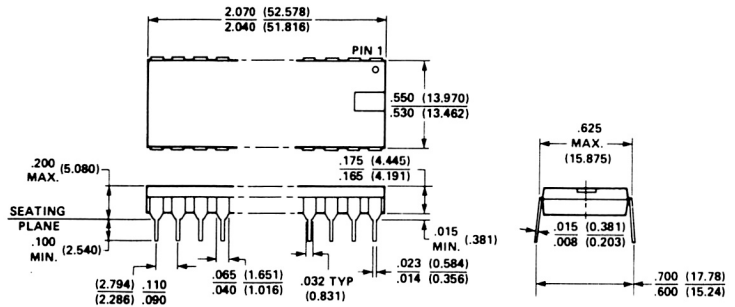
**28-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE D**



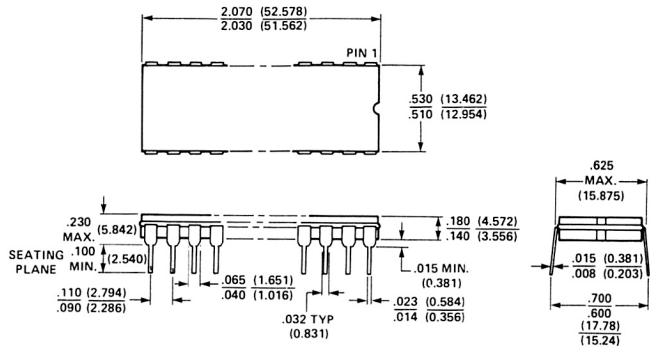
**28-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C**



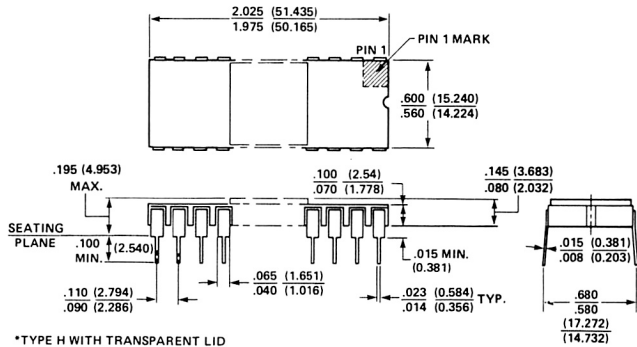
**40-LEAD PLASTIC DUAL IN-LINE  
PACKAGE TYPE P**



**40-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE D**



**40-LEAD HERMETIC DUAL IN-LINE  
PACKAGE TYPE C OR H\***



1. The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that proper record-keeping is essential for the efficient operation of any business or organization. The text outlines the various methods used to collect and organize data, including the use of ledgers, journals, and other accounting systems. It also discusses the importance of regular audits and the role of internal controls in ensuring the integrity of the financial records.

2. The second part of the document focuses on the analysis and interpretation of the recorded data. It describes the various techniques used to identify trends, patterns, and anomalies in the financial information. This section also discusses the importance of comparing the recorded data with budgeted figures and industry standards. The text provides examples of how this analysis can be used to make informed decisions about the future operations of the organization.

3. The third part of the document discusses the reporting of financial information. It outlines the various types of financial statements that are typically prepared, including the balance sheet, income statement, and cash flow statement. It also discusses the importance of providing clear and concise explanations of the data presented in these statements. The text provides examples of how to format and present financial information in a way that is easy to understand and interpret.

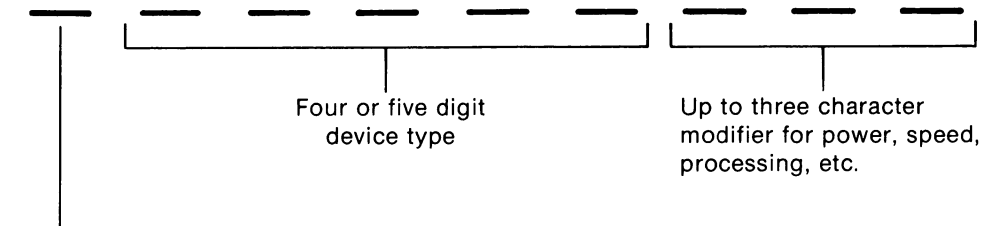
4. The fourth part of the document discusses the use of financial information for decision-making. It describes the various ways in which financial data can be used to evaluate the performance of different departments or projects. It also discusses the importance of using financial information to identify areas for improvement and to develop strategies for future growth. The text provides examples of how financial data can be used to make informed decisions about the allocation of resources and the overall direction of the organization.

5. The final part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that proper record-keeping is essential for the efficient operation of any business or organization. The text outlines the various methods used to collect and organize data, including the use of ledgers, journals, and other accounting systems. It also discusses the importance of regular audits and the role of internal controls in ensuring the integrity of the financial records.

## APPENDIX 2

### ORDERING INFORMATION

Semiconductor components are ordered as follows:



Package Type

**B** — Hermetic Package, Type B

**C** — Hermetic Package, Type C

**D** — Hermetic Package, Type D

**H** — Hermetic Package, Type C with  
Transparent Window for EPROMs

**P** — Plastic Package

#### Examples

**P5101L**      CMOS 256 x 4 RAM, low power selection  
plastic package

**D8048**      8048 Microcomputer, hermetic package Type D

The latest Intel price book should be consulted for availability of various options.

# THE UNIVERSITY OF CHICAGO LIBRARY

Acquired from the University of Chicago Library

Author	Title	Date	Volume	Page	Notes
The University of Chicago	Library				

- 1. The University of Chicago
- 2. The University of Chicago
- 3. The University of Chicago
- 4. The University of Chicago
- 5. The University of Chicago

THE UNIVERSITY OF CHICAGO  
LIBRARY  
1000 S. MICHIGAN AVE.  
CHICAGO, ILL. 60607

This book is the property of the University of Chicago Library and is loaned to you for your personal use only. It is not to be resold, lent, or otherwise disposed of without the written permission of the University of Chicago Library.

## NOTES

## NOTES



## NOTES

## NOTES

## NOTES

## NOTES

## NOTES

## NOTES

## NOTES

## NOTES







3055 Bowers Avenue  
Santa Clara, California 95051  
Tel: (408) 246-7501  
TWX: 910-338-0026  
TELEX: 34-6372

## SALES AND MARKETING OFFICES

### U.S. AND CANADIAN SALES OFFICES

#### ALABAMA

Glen White Associates  
7844 Horseshoe Trail  
Huntsville 35802  
Tel: (205) 883-9394

#### ARIZONA

Sales Engineering, Inc.  
7155 E. Thomas Road, No. 6  
Scottsdale 85252  
Tel: (602) 994-3230  
TWX: 910-950-1288

#### CALIFORNIA

Intel Corp.\*  
990 E. Arques Ave.  
Suite 112  
Sunnyvale 94086  
Tel: (408) 738-3870  
TWX: 910-339-9279  
Tel: (408) 738-0255

Mac-I  
P.O. Box 1420  
Cupertino 95014  
Tel: (408) 257-9880

Earle Associates, Inc.  
4433 Convey Street  
Suite A  
San Diego 92111  
Tel: (714) 278-5441  
TWX: 910-335-1585

Intel Corp.\*  
1651 East 4th Street  
Suite 228  
Santa Ana 92701  
Tel: (714) 835-9642  
TWX: 910-595-1114

#### COLORADO

Intel Corp.  
12075 East 45th Avenue  
Suite 310  
Denver 80239  
Tel: (303) 373-9920  
TWX: 910-932-0322

#### CONNECTICUT

Intel Corp.  
8 Mill Plain Road  
Danbury 06810  
Tel: (203) 792-8365

### EUROPEAN MARKETING OFFICES

#### BELGIUM

Intel International\*  
Rue du Moulin à Papier  
51-Boile 1  
B-1160 Brussels  
Tel: (02) 660 30 10  
TELEX: 24814

#### FLORIDA

Intel Corp.  
2020 W. McNab Road, Suite 104  
Ft. Lauderdale 33309  
Tel: (305) 971-7200  
TWX: 910-956-9407  
Intel Corp.  
5151 Adamson Street, Suite 200-3  
Orlando 32804  
Tel: (305) 628-2393  
TWX: 810-853-9219

#### ILLINOIS

Intel Corp.\*  
1000 Jorie Boulevard  
Suite 224  
Oakbrook 60521  
Tel: (312) 325-9510  
TWX: 910-651-5881

#### IOWA

Technical Representatives, Inc.  
1703 Hillside Drive  
Cedar Rapids  
Tel: (319) 396-5662

#### KANSAS

Technical Representatives, Inc.  
801 Clairborne  
Olathe 66061  
Tel: (913) 782-1177  
TWX: 910-749-6412

#### MARYLAND

Glen White Associates  
57 West Timonium Road  
Timonium 21093  
Tel: (301) 252-7742

#### MASSACHUSETTS

Intel Corp.\*  
57 West Timonium Road  
Suite 307  
Timonium 21093  
Tel: (301) 252-7742  
TWX: 710-232-1772  
Intel Corp.\*  
187 Billerica Road, Suite 14A  
Chelmsford 01824  
Tel: (617) 861-1136  
TWX: 710-343-6333

#### MICHIGAN

Intel Corp.  
725 South Adams Road  
Suite 288  
Birmingham 48011  
Tel: (313) 642-7018  
TWX: 910-420-1212  
TELEX: 2 31143

#### MINNESOTA

Intel Corp.  
675 Southgate Office Plaza  
5001 West 80th Street  
Bloomington 55437  
Tel: (612) 835-6722  
TWX: 910-576-2867

#### MISSOURI

Technical Representatives, Inc.  
Trade Center Bldg.  
300 Brookside Drive, Suite 108  
Hazelwood 63042  
Tel: (314) 731-5200  
TWX: 910-762-0618

#### NEW JERSEY

Intel Corp.  
2 Kilmer Road  
Edison 08817  
Tel: (201) 985-9100  
TWX: 710-480-6238

#### NEW YORK

Intel Corp.\*  
8901 Jericho Turnpike  
Syosset 11791  
Tel: (516) 364-9860  
TWX: 510-221-2198  
Intel Corp.  
474 Thurston Road  
Rochester, N.Y. 14619  
Tel: (716) 328-7340  
TWX: 510-253-3841

#### NEW YORK (cont.)

T-Squared  
3522 James Street  
Syracuse 13206  
Tel: (315) 463-8592  
TWX: 710 541-0554  
T-Squared  
640 Craig Road  
P.O. Box W  
Pittsford 14534  
Tel: (716) 381-2551  
TELEX: 97-8289  
Intel Corp.  
55 Market Street  
Poughkeepsie, New York 12601  
Tel: (914) 473-2303  
TWX: 510-248-0060

#### NORTH CAROLINA

Glen White Associates  
913 Plateau Lane  
Raleigh 27609  
Tel: (919) 876-5617

#### OHIO

Intel Corp.\*  
8312 North Main Street  
Dayton 45415  
Tel: (513) 890-5350  
TELEX: 288-004  
Intel Corp.\*  
28250 Euclid Ave.  
Suite 531F  
Euclid 44132  
Tel: (216) 289-0101

#### PENNSYLVANIA

Intel Corp.\*  
520 Pennsylvania Ave.  
Fort Washington 19034  
Tel: (215) 542-9444  
TWX: 510-661-0709

#### TENNESSEE

Glen White Associates  
206 Chickasaw Drive  
Johnson City 37601  
Tel: (615) 928-0184

#### TEXAS

Evans & McDowell Associates  
Suite 405  
Dallas 75231  
Tel: (214) 238-7157  
TWX: 910-867-4763  
Evans & McDowell Associates  
8610 Harwin Avenue, Suite 125  
Houston 77036  
Tel: (713) 783-2900  
Intel Corp.\*  
6350 L.B.J. Freeway  
Suite 178  
Dallas 75240  
Tel: (214) 661-8829  
TWX: 910-860-5487

#### VIRGINIA

Glen White Associates  
P.O. Box 1104  
Lynchburg 24505  
Tel: (804) 846-4624  
WASHINGTON  
E.S./Chase Co.  
P.O. Box 60093  
Seattle 98108  
Tel: (206) 762-4824  
TWX: 910-444-2298

#### CANADA

Intel Corp.  
70 Chamberlain Ave.  
Ottawa, Ontario K1S 1V9  
Tel: (613) 232-8576  
TELEX: 053-4419  
Multitek, Inc.  
4 Barren Street  
Ottawa, Ontario K2J 1G2  
Tel: (613) 825-4695  
TELEX: 053-4585

#### FRANCE

Intel Corporation, S.A.R.L.\*  
74, Rue D'Arcueil  
Siliç 223  
45928 Rangis Cedex  
France  
Tel: (01) 687 22 21  
TELEX: 270475

#### SCANDINAVIA

Intel Scandinavia A/S\*  
Lyngbyvej 32 2nd Floor  
DK-2100 Copenhagen East  
Denmark  
Tel: (01) 18 20 00  
TELEX: 19567  
Intel Sweden AB\*  
Box 86  
S-1612 Vällingby 1  
Sweden  
Tel: (08) 37 53 70  
TELEX: 13164 (ABCENT)

#### ENGLAND

Intel Corporation (U.K.) Ltd.\*  
Broadfield House  
4 Between Towns Road  
Cowley, Oxford OX4 3NB  
Tel: (0865) 77 14 31  
TELEX: 837203  
Intel Corporation (U.K.) Ltd.  
46-50 Beam Street  
Nantwich, Cheshire CW5 5LJ  
Tel: (0270) 62 65 60  
TELEX: 36620

#### GERMANY

Intel Semiconductor GmbH\*  
Seidstrasse 27  
8000 Muenchen 2  
Tel: (089) 55 81 41  
TELEX: 523 177  
Intel Semiconductor GmbH  
D-6272 Niederrhausen  
Wiesenweg 26  
Tel: (06127) 2314  
TELEX: 04186183  
Intel Semiconductor GmbH  
D-7000 Stuttgart 80  
Emstaldenstrasse 17  
Tel: (0711) 7351506  
TELEX: 7255346

### ORIENT MARKETING OFFICES

#### JAPAN

Intel Japan Corporation\*  
Flower Hill-Shinmachi East Bldg.  
1-23-9, Shinmachi, Setagaya-ku  
Tokyo 154  
Tel: (03) 426-9261  
TELEX: 781-28426

#### HONG KONG

Q1 (Far East) Ltd.  
Tak Yan Commercial Bldg, 8th floor  
30-32 D'Aguilar Street, Central  
Hong Kong  
Tel: 5-260311  
TELEX: 83138 JADE HX

#### TAIWAN

Taiwan Automation Co.\*  
6th Floor, 18-1, Lane 14  
Chih-Lin Road  
Taipei  
Tel: (02) 551726-9  
TELEX: 11942 TAIAUTO

#### TAIWAN (cont.)

Asionics-Taiwan, Inc.  
205 Pa-Teh Road, Section 4  
Taipei  
Tel: 75 55 82  
TELEX: 22158 Asionics

### INTERNATIONAL DISTRIBUTORS

#### AUSTRALIA

A. J. Ferguson (Adelaide) PTY. Ltd.  
44 Prospect Rd.  
Prospect 5082  
South Australia  
Tel: 269-1244  
TELEX: 82635

#### AUSTRIA

Bacher Elektronische Geräte GmbH  
Maidinger Hauptstrasse 78  
A 1120 Vienna  
Tel: (0222) 83 63 96  
TELEX: (01) 1532

#### BELGIUM

Inelco Belgium S.A.  
Avenue Val Duchesne, 3  
B-1160 Brussels  
Tel: (02) 660 00 12  
TELEX: 25441

#### DENMARK

Scandinavian Semiconductor  
Supply A/S  
Nannasgade 18  
DK-2200 Copenhagen N  
Tel: (01) 93 50 90  
TELEX: 19037

#### FINLAND

Oy Fintronic AB  
Astekinkatu 2C.  
SF 00520  
Helsinki 52  
Tel: (90) 664 451  
TELEX: 12426

#### FRANCE

Tekelec Airtronic  
Cite des Bruyeres  
Rue Carle Vermet  
92310 Sevres  
Tel: (1) 027 75 35  
TELEX: 250997

#### GERMANY

Alfred Neys Enatechnik GmbH  
Schillerstrasse 14  
D-2085 Quickborn-Hamburg  
Tel: (04108) 6121  
TELEX: 02-13530  
Electronic 2000 Vertriebs GmbH  
Neumarkter Strasse 75  
D-8000 Muenchen 80  
Tel: (089) 434061  
TELEX: 48426  
Jermyn GmbH  
Postfach 1148  
D-6277 Kumburg  
Tel: (06434) 6005  
TELEX: 48426

#### HONG KONG

ASTEC International  
Keystone House, 2nd Floor  
Hankow Road, Kowloon  
Tel: 5-867750  
TELEX: 74899 ASCOM

#### ISRAEL

Eastronics Ltd.\*  
11 Rozania Street  
P.O. Box 39300  
Tel-Aviv  
Tel: 475151  
TELEX: 33638

#### ITALY

Eledra 3S S.P.A.  
Viale Elvezio 18  
20154 Milan,  
Tel: (02) 3493041  
TELEX: 39332  
Eledra 3S S.P.A.  
Via Giuseppe Valmarana, 63  
00139 Rome, Italy  
Tel: (06) 81 27 290 - 81 27 324

#### JAPAN

Pan Electron  
No. 1 Higashikita-Machi  
Midori-Ku, Yokohama 226  
Tel: (045) 471-8811  
TELEX: 781-4773  
Ryoyo Electric Corp.  
Konwa Bldg.  
1-12-22, Tsukiji, 1-Chome  
Chuo-Ku, Tokyo 104  
Tel: (03) 543-7711

#### NETHERLANDS

Inelco Nederland  
AFD Elektroniek  
Joan Muskensweg 22  
NL-1006 Amsterdam  
Tel: (020) 934824  
TELEX: 14622  
NORWAY  
Nordisk Elektronik (Norge) A/S  
Postboks 1  
N-0512 Oslo 2  
Tel: (02) 55 38 93  
TELEX: 16963

#### SOUTH AFRICA

Electronic Building Elements  
P.O. Box 4609  
Pretoria  
Tel: 78 92 21  
TELEX: 30181  
SPAIN  
Inierface  
Ronda San Pedro 22  
Barcelona 10  
Tel: 30\* 78 51

#### SWEDEN

Nordisk Elektronik AB  
Fack  
S-10380 Stockholm 7  
Tel: (08) 248340  
TELEX: 10547

#### SWITZERLAND

Industrie AG  
Gemensstrasse 2  
Postfach 80 - 21190  
CH-8021 Zurich  
Tel: (01) 60 22 30  
TELEX: 56788

#### UNITED KINGDOM

Rapid Recall, Ltd.  
11-15 Betterton Street  
Drury Lane  
London WC2H 9BS  
Tel: (01) 379-6741  
TELEX: 28752  
G.E.C. Semiconductors Ltd.  
East Lane  
Wembley HA9 7PP  
Middlesex  
Tel: (01) 904-9303  
TELEX: 923429  
Jermyn Industries  
Bexley, Sevenoaks Road  
Sevenoaks, Kent.  
Tel: (0732) 51174  
TELEX: 95143

\*Field Application Location



## U.S. AND CANADIAN DISTRIBUTORS

3065 Bowers Avenue  
Santa Clara, California 95051  
Tel: (408) 246-7501  
TWX: 910-338-0026  
TELEX: 34-6372

### U.S. AND CANADIAN DISTRIBUTORS

#### ALABAMA

IHamilton/Avnet Electronics  
805 Osar Drive NW  
Huntsville 35805  
Tel: (205) 533-1170

#### ARIZONA

Hamilton/Avnet Electronics  
2615 South 21st Street  
Phoenix 85034  
Tel: (602) 275-7851  
Liberty/Arizona  
3130 N. 27th Avenue  
Phoenix 85107  
Tel: (602) 257-1272  
TELEX: 910-951-4282

#### CALIFORNIA

IHamilton/Avnet Electronics  
575 E. Middlefield Road  
Mountain View 94040  
Tel: (415) 981-7000  
IHamilton/Avnet Electronics  
8917 Complex Drive  
San Diego 92123  
Tel: (714) 279-2421  
IHamilton Electro Sales  
10312 W. Washington Boulevard  
Culver City 90230  
Tel: (213) 558-2121  
ICramer/San Francisco  
720 Palomar Avenue  
Sunnyvale 94086  
Tel: (408) 739-3011  
ICramer/Los Angeles  
1720 Daimler Street  
Irvine 92705  
Tel: (714) 979-3000  
ILiberty Electronics  
124 Maryland Street  
El Segundo 90245  
Tel: (213) 322-8100  
Tel: (714) 638-7601  
TWX: 910-348-7140  
Liberty/San Diego  
8248 Mercury Court  
San Diego 92111  
Tel: (714) 565-2171  
Tel: (619) 335-1590  
Elmar Electronics  
2288 Charleston Road  
Mountain View 94040  
Tel: (415) 961-3611  
TELEX: 910-379-6437

#### COLORADO

Cramer/Denver  
5465 E. Evans Pl. at Hudson  
Denver 80222  
Tel: (303) 758-2100  
Elmar/Denver  
6777 E. 50th Avenue  
Commerce City 80022  
Tel: (303) 287-9611  
TWX: 910-936-0770  
IHamilton/Avnet Electronics  
5921 No. Broadway  
Denver 80216  
Tel: (303) 534-1212

#### CONNECTICUT

Cramer/Connecticut  
35 Dodge Avenue  
North Haven 06473  
Tel: (203) 239-5641  
Hamilton/Avnet Electronics  
643 Danbury Road  
Georgetown 06829  
Tel: (203) 762-0361

#### FLORIDA

Cramer/E.W. Hollywood  
4035 No. 29th Avenue  
Hollywood 33020  
Tel: (305) 923-8181  
Hamilton/Avnet Electronics  
4020 No. 29th Ave.  
Hollywood 33021  
Tel: (305) 925-5401  
ICramer/E.W. Orlando  
345 No. Graham Ave.  
Orlando 32814  
Tel: (305) 894-1511

#### GEORGIA

Cramer/E.W. Atlanta  
3923 Calcutt Industrial Center  
Atlanta 30340  
Tel: (404) 448-9050  
Hamilton/Avnet Electronics  
6700 185. Access Road, Suite 2B  
Norcross 30071  
Tel: (404) 448-0800

#### ILLINOIS

ICramer/Chicago  
1911 So. Busse Rd.  
Mt. Prospect 60056  
Tel: (312) 593-8230  
IHamilton/Avnet Electronics  
3901 No. 25th Ave.  
Schiller Park 60176  
Tel: (312) 678-6310

#### INDIANA

Pioneer/Indiana  
6408 Castleplace Drive  
Indianapolis 46250  
Tel: (317) 649-7300  
Sheridan Sales Co.  
8790 Purdue Road  
Indianapolis 46268  
Tel: (317) 297-3148

#### KANSAS

Hamilton/Avnet Electronics  
37 Lenexa Industrial Center  
9900 Plumb Road  
Lenexa 66215  
Tel: (913) 888-8900

#### MARYLAND

Cramer/E.W. Baltimore  
7235 Standard Drive  
Hanover 21076  
Tel: (301) 796-5790  
ICramer/E.W. Washington  
65021 Industrial Drive  
Gaithersburg 20760  
Tel: (301) 948-0110  
Hamilton/Avnet Electronics  
7235 Standard Drive  
Hanover 21076  
Tel: (301) 796-5000  
Pioneer/Washington  
11000 Galtier Road  
Gaithersburg 20760  
Tel: (301) 948-0710  
TWX: 710-828-0945

#### MASSACHUSETTS

ICramer Electronics Inc.  
85 Wells Avenue  
Newton 02159  
Tel: (617) 969-7700  
Hamilton/Avnet Electronics  
100 E. Commerce Way  
Woburn 01801  
Tel: (617) 933-8000

#### MICHIGAN

Sheridan Sales Co.  
24543 Indoplex Drive  
Farmington Hills 48024  
Tel: (313) 477-3800  
IPioneer/Michigan  
13485 Stamford  
Livonia 48150  
Tel: (313) 728-8500  
IHamilton/Avnet Electronics  
32487 Schoolcraft Road  
Livonia 48150  
Tel: (313) 522-4700  
TWX: 810-242-8775

#### MINNESOTA

Industrial Components  
5280 West 74th Street  
Minneapolis 55435  
Tel: (612) 831-2666  
Cramer/Bonn  
7275 Bush Lane Road  
Edina 55435  
Tel: (612) 835-7811  
Hamilton/Avnet Electronics  
6852 Washington Avenue So.  
Edina 55435  
Tel: (612) 941-3801

#### MISSOURI

IHamilton/Avnet Electronics  
354 Brooks Lane  
Hazelwood 63042  
Tel: (314) 731-1144

#### NEW JERSEY

Cramer/Pennsylvania, Inc.  
12 Springdale Road  
Cherry Hill Industrial Center  
Cherry Hill 08003  
Tel: (609) 424-5993  
TWX: 710-896-8908  
Hamilton/Avnet Electronics  
218 Little Falls Road  
Cedar Grove 07009  
Tel: (201) 239-0800  
TWX: 710-994-5787  
Cramer/New Jersey  
No. 1 Barrett Avenue  
Moonachie 07074  
Tel: (201) 935-5600

#### NEW JERSEY (cont.)

IHamilton/Avnet Electronics  
113 Galtier Drive  
East Gate Industrial Park  
Mt. Laurel 08057  
Tel: (609) 234-2133  
TWX: 710-897-1405

#### NEW MEXICO

Hamilton/Avnet Electronics  
2450 Baylor Drive, S.E.  
Albuquerque 87119  
Tel: (505) 765-1500  
Cramer/New Mexico  
137 Vermont, N.E.  
Albuquerque 87108  
Tel: (505) 265-5767

#### NEW YORK

Cramer/Rochester  
3000 Winton Road South  
Rochester 14623  
Tel: (716) 275-0300  
IHamilton/Avnet Electronics  
167 Clay Road  
Rochester 14623  
Tel: (716) 442-7820  
ICramer/Syracuse  
6718 Joy Road  
East Syracuse 13057  
Tel: (315) 437-6671  
IHamilton/Avnet Electronics  
6500 Joy Road  
E. Syracuse 13057  
Tel: (315) 437-2642  
ICramer/Long Island  
29 Oran Avenue  
Hempstead, L.I. 11787  
Tel: (516) 231-5800  
TWX: 510-227-9863  
IHamilton/Avnet Electronics  
70 State Street  
Westbury, L.I. 11590  
Tel: (516) 333-5800  
TWX: 510-222-8237

#### NORTH CAROLINA

Cramer Electronics  
938 Burke Street  
Winston-Salem 27102  
Tel: (919) 725-8711  
Pioneer/Carolina  
2906 Baltic Avenue  
Greensboro 27406  
Tel: (919) 273-4444  
TWX: 510-925-1114

#### OHIO

IHamilton/Avnet Electronics  
118 Westpark Road  
Dayton 45459  
Tel: (513) 433-0610  
TWX: 810-450-2531  
IPioneer/Dayton  
1900 Troy Street  
Dayton 45404  
Tel: (513) 236-9900  
ISheridan Sales Co.  
10 Knollcrest Drive  
Cincinnati 45222  
Tel: (513) 761-5432  
TWX: 810-461-2670  
IPioneer/Cleveland  
4800 E. 131st Street  
Cleveland 44105  
Tel: (216) 587-3600  
IHamilton/Avnet Electronics  
761 Beta Drive  
Cleveland 44143  
Tel: (216) 461-1400  
Sheridan Sales Co.  
23224 Commerce Park Road  
Beachwood 44122  
Tel: (216) 831-0130  
Sheridan Sales Co.  
Shilon Building, Suite 250  
5045 North Main Street  
Dayton 45405  
Tel: (513) 277-8911

#### OKLAHOMA

Component Specialties, Inc.  
7920 E. 40th Street  
Tulsa 74145  
Tel: (918) 864-2820

#### OREGON

Almac/Stroum Electronics  
4475 S.W. Scholla Ferry Rd.  
Portland 97225  
Tel: (503) 292-2534

#### PENNSYLVANIA

Sheridan Sales Co.  
1717 Penn Avenue, Suite 5009  
Pittsburgh 15221  
Tel: (412) 244-1640  
Pioneer/Pittsburgh  
560 Alpha Drive  
Pittsburgh 15238  
Tel: (412) 782-2300

#### PENNSYLVANIA (cont.)

Pioneer/Delaware  
203 Wilmer Road  
Horsham 19044  
Tel: (215) 647-5710  
TWX: 510-665-8778

#### TEXAS

Cramer Electronics  
13740 Midway Road  
Dallas 75240  
Tel: (214) 661-9300  
IHamilton/Avnet Electronics  
4445 Sigma Road  
Dallas 75240  
Tel: (214) 661-8661  
IHamilton/Avnet Electronics  
1218 W. Clay  
Houston 77019  
Tel: (713) 526-4661  
Component Specialties, Inc.  
10907 Shady Trail, Suite 101  
Dallas 75220  
Tel: (214) 357-6511  
IComponent Specialties, Inc.  
7313 Ashcroft Street  
Houston 77036  
Tel: (713) 771-7237

#### UTAH

Hamilton/Avnet Electronics  
647 W. Billings Road  
Salt Lake City 84119  
Tel: (801) 862-8451

#### WASHINGTON

IHamilton/Avnet Electronics  
13407 Northrup Way  
Bellevue 98005  
Tel: (206) 746-8750  
IAlmac/Stroum Electronics  
5811 Sixth Ave. South  
Seattle 98108  
Tel: (206) 763-2300

### CANADA

#### ALBERTA

L.A. Varah Ltd.  
4742 14th Street N.E.  
Calgary T2E 6L7  
Tel: (403) 276-8818  
Telex: 13 825 89 77

#### BRITISH COLUMBIA

L.A. Varah Ltd.  
2077 Alberta Street  
Vancouver V6Y 1C4  
Tel: (604) 873-3211  
TWX: 810-929-1068  
Telex: 04 53167

#### ONTARIO

Hamilton/Avnet Electronics  
6291-16 Dorman Road  
Mississauga L4V 1H2  
Tel: (416) 677-7432  
TWX: 810-492-8667

#### Hamilton/Avnet Electronics

1735 Courtwood Cresc.  
Ottawa K2C 2B4  
Tel: (613) 226-1700  
TWX: 610 562-1906

#### Zenitronics

141 Catherine Street  
Ottawa, Ontario K2P 1C3  
Tel: (613) 238-6411

#### Zenitronics

185 Bridgeland Avenue  
Toronto, Ontario M6A 1Z3  
Tel: (416) 787-1271  
Telex: 02-021694

#### QUEBEC

Hamilton/Avnet Electronics  
2670 Paulus  
St. Laurent H4S 1G2  
Tel: (514) 331-6443

#### L.A. Varah Ltd.

1832 King Edward Street  
Winnipeg R2R 0N1  
Tel: (204) 633-6190  
Telex: 07-53365

#### Zenitronics

8148 Montview Road  
Town of Mount Royal, Montreal  
Quebec H4P 2L7  
Tel: (514) 735-5361  
Telex: 05-827535

#### MANITOBA

L.A. Varah Ltd.  
153 Corbett Drive  
Winnipeg R2Y 1V4  
Tel: (204) 889-9807

#### L.A. Varah Ltd.

1832 King Edward Street  
Winnipeg R2R 0N1  
Tel: (204) 633-6190  
Telex: 07-53365

# HEXADECIMAL INSTRUCTION CODES

## ACCUMULATOR

• ADD A,R <sub>r</sub>	6•
• ADD A,@R0	60
R1	61
• ADD A,#data	03
• ADDC A,R <sub>r</sub>	7•
• ADDC A,@R0	70
R1	71
• ADDC A,#data	13
ANL A,R <sub>r</sub>	5•
ANL A,@R0	50
R1	51
ANL A,#data	53
ORL A,R <sub>r</sub>	4•
ORL A,@R0	40
R1	41
ORL A,#data	43
XRL A,R <sub>r</sub>	D•
XRL A,@R0	D0
R1	D1
XRL A,#data	D3
INC A	17
DEC A	07
CLR A	27
CPL A	37
RL A	E7
• RLC A	F7
RR A	77
• RRC A	67
• DA A	57
SWAP A	47

## DATA MOVES

MOV A,R <sub>r</sub>	F•
MOV A,@R0	F0
R1	F1
MOV A,#data	23
MOV R,A <sub>r</sub>	A•
MOV @R0,A	A0
R1,A	A1
MOV R <sub>r</sub> ,#data	B•
MOV @R0,#data	B0
R1,#data	B1
XCH A,R <sub>r</sub>	2•
XCH A,@R0	20
R1	21
XCHD A,@R0	30
R1	31
MOV A,PSW	C7
• MOV PSW,A	D7
MOVX A,@R0	80
R1	81
MOVX @R0,A	90
R1,A	91
MOVP3 A,@A	E3
MOVP A,@A	A3

## REGISTER

INC R <sub>r</sub>	1•
DEC R <sub>r</sub>	C•
INC @R0	10
R1	11
DJNZ R <sub>r</sub> , addr	E•

## FLAGS

• CLR C	97
• CPL C	A7
CLR F0	85
CPL F0	95
CLR F1	A5
CPL F1	B5

## BRANCH

JMP addr	†4
JMPP @A	B3
DJNZ R <sub>r</sub> ,addr	E•
JC addr	F6
JNC addr	E6
JZ addr	C6
JNZ addr	96
JT0 addr	36
JNT0 addr	26
JT1 addr	56
JNT1 addr	46
JF0 addr	B6
JF1 addr	76
JTF addr	16
JNI addr	86
JB0 addr	12
JB1 addr	32
JB2 addr	52
JB3 addr	72
JB4 addr	92
JB5 addr	B2
JB6 addr	D2
JB7 addr	F2

## TIMER

MOV A,T	42
MOV T,A	62
STRT T	55
STRT CNT	45
STOP TCNT	65
EN TCNTI	25
DIS TCNTI	35

## CONTROL

EN I	05
DIS I	15
SEL RB0	C5
SEL RB1	D5
SEL MB0	E5
SEL MB1	F5
ENT0 CLK	75

## SUBROUTINE

CALL addr	†4
RET	83
RETR	93

## NO OP

NOP	00
-----	----

## INPUT/OUTPUT:

IN A,P1	09
OUTL P1,A	39
ANL P1,#data	99
ORL P1,#data	89
IN A, P2	0A
OUT L P2, A	3A
ANL P2, #data	9A
ORL P2,#data	8A
INS A, BUS	08
OUTL BUS, A	02
ANL BUS, #data	98
ORL BUS, #data	88
MOVD A,Pp	0:
MOVD Pp,A	3:
ANLD Pp,A	9:
ORLD Pp,A	8:

• = Carry Flag Affected

\* = See Table 1

⋮ = See Table 2

† = See Table 3

## Join the MCS-48™ Mailing List

Help us keep you up to date with the latest MCS-48 product information and application notes by filling in the following information.

Name \_\_\_\_\_

Company \_\_\_\_\_

Title \_\_\_\_\_

Mail Stop \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_

- ☐ Design Engineer
- ☐ Engineering Manager
- ☐ Engineering Support
- ☐ Manufacturing
- ☐ Purchasing
- ☐ Hobbyist
- ☐ Other \_\_\_\_\_

### Number of Chips to be used per system \_\_\_\_\_

- |                                      |                               |                               |                               |                               |
|--------------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
| <input type="checkbox"/> 8048        | <input type="checkbox"/> 8355 | <input type="checkbox"/> 8212 | <input type="checkbox"/> 8214 | <input type="checkbox"/> 8708 |
| <input type="checkbox"/> 8748        | <input type="checkbox"/> 8755 | <input type="checkbox"/> 8251 | <input type="checkbox"/> 8216 | <input type="checkbox"/> 8316 |
| <input type="checkbox"/> 8035        | <input type="checkbox"/> 8155 | <input type="checkbox"/> 8253 | <input type="checkbox"/> 8259 | <input type="checkbox"/> 8101 |
| <input type="checkbox"/> 8243        | <input type="checkbox"/> 8205 | <input type="checkbox"/> 8255 | <input type="checkbox"/> 8279 | <input type="checkbox"/> 8111 |
| <input type="checkbox"/> Other _____ |                               |                               |                               |                               |

### System Requirements

- |                |                               |                              |                              |                               |                               |
|----------------|-------------------------------|------------------------------|------------------------------|-------------------------------|-------------------------------|
| Program Memory | <input type="checkbox"/> 1Kx8 | <input type="checkbox"/> 2K  | <input type="checkbox"/> 3K  | <input type="checkbox"/> 4K   | <input type="checkbox"/> More |
| Data Memory    | <input type="checkbox"/> 64x8 | <input type="checkbox"/> 256 | <input type="checkbox"/> 512 | <input type="checkbox"/> 1K   | <input type="checkbox"/> More |
| Input Lines    | <input type="checkbox"/> 8    | <input type="checkbox"/> 16  | <input type="checkbox"/> 24  | <input type="checkbox"/> 32   | <input type="checkbox"/> More |
| Output Lines   | <input type="checkbox"/> 16   | <input type="checkbox"/> 24  | <input type="checkbox"/> 32  | <input type="checkbox"/> 40   | <input type="checkbox"/> More |
| Timers         | <input type="checkbox"/> 1    | <input type="checkbox"/> 2   | <input type="checkbox"/> 3   | <input type="checkbox"/> More |                               |

### Product Description

- |  |   |
|--|---|
| <input type="checkbox"/> Military        | <input type="checkbox"/> Terminal           |
| <input type="checkbox"/> Consumer        | <input type="checkbox"/> Process Controller |
| <input type="checkbox"/> Industrial      | <input type="checkbox"/> Machine Controller |
| <input type="checkbox"/> Data Processing | <input type="checkbox"/> Instrument         |
| <input type="checkbox"/> Other _____     | <input type="checkbox"/> Test Equipment     |
|  | <input type="checkbox"/> In-house Equipment |
|  | <input type="checkbox"/> Other _____        |

Production date \_\_\_\_\_ Systems/Month \_\_\_\_\_

Name \_\_\_\_\_

Company \_\_\_\_\_

Title \_\_\_\_\_

Mail Stop \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_

- ☐ Design Engineer
- ☐ Engineering Manager
- ☐ Engineering Support
- ☐ Manufacturing
- ☐ Purchasing
- ☐ Hobbyist
- ☐ Other \_\_\_\_\_

### Number of Chips to be used per system \_\_\_\_\_

- |                                      |                               |                               |                               |                               |
|--------------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
| <input type="checkbox"/> 8048        | <input type="checkbox"/> 8355 | <input type="checkbox"/> 8212 | <input type="checkbox"/> 8214 | <input type="checkbox"/> 8708 |
| <input type="checkbox"/> 8748        | <input type="checkbox"/> 8755 | <input type="checkbox"/> 8251 | <input type="checkbox"/> 8216 | <input type="checkbox"/> 8316 |
| <input type="checkbox"/> 8035        | <input type="checkbox"/> 8155 | <input type="checkbox"/> 8253 | <input type="checkbox"/> 8259 | <input type="checkbox"/> 8101 |
| <input type="checkbox"/> 8243        | <input type="checkbox"/> 8205 | <input type="checkbox"/> 8255 | <input type="checkbox"/> 8279 | <input type="checkbox"/> 8111 |
| <input type="checkbox"/> Other _____ |                               |                               |                               |                               |

### System Requirements

- |                |                               |                              |                              |                               |                               |
|----------------|-------------------------------|------------------------------|------------------------------|-------------------------------|-------------------------------|
| Program Memory | <input type="checkbox"/> 1Kx8 | <input type="checkbox"/> 2K  | <input type="checkbox"/> 3K  | <input type="checkbox"/> 4K   | <input type="checkbox"/> More |
| Data Memory    | <input type="checkbox"/> 64x8 | <input type="checkbox"/> 256 | <input type="checkbox"/> 512 | <input type="checkbox"/> 1K   | <input type="checkbox"/> More |
| Input Lines    | <input type="checkbox"/> 8    | <input type="checkbox"/> 16  | <input type="checkbox"/> 24  | <input type="checkbox"/> 32   | <input type="checkbox"/> More |
| Output Lines   | <input type="checkbox"/> 16   | <input type="checkbox"/> 24  | <input type="checkbox"/> 32  | <input type="checkbox"/> 40   | <input type="checkbox"/> More |
| Timers         | <input type="checkbox"/> 1    | <input type="checkbox"/> 2   | <input type="checkbox"/> 3   | <input type="checkbox"/> More |                               |

### Product Description

- |  |   |
|--|---|
| <input type="checkbox"/> Military        | <input type="checkbox"/> Terminal           |
| <input type="checkbox"/> Consumer        | <input type="checkbox"/> Process Controller |
| <input type="checkbox"/> Industrial      | <input type="checkbox"/> Machine Controller |
| <input type="checkbox"/> Data Processing | <input type="checkbox"/> Instrument         |
| <input type="checkbox"/> Other _____     | <input type="checkbox"/> Test Equipment     |
|  | <input type="checkbox"/> In-house Equipment |
|  | <input type="checkbox"/> Other _____        |

Production date \_\_\_\_\_ Systems/Month \_\_\_\_\_

FIRST CLASS  
PERMIT NO. 621  
SANTA CLARA  
CA. 95051

No Postage Necessary if Mailed Inside the United States

**BUSINESS REPLY MAIL**

Postage Will Be Paid By

INTEL CORPORATION  
Microcomputer Division  
3065 Bowers Avenue  
Santa Clara, CA 95051

Attn: MCS-48™ Components

FIRST CLASS  
PERMIT NO. 621  
SANTA CLARA  
CA. 95051

No Postage Necessary if Mailed Inside the United States

**BUSINESS REPLY MAIL**

Postage Will Be Paid By

INTEL CORPORATION  
Microcomputer Division  
3065 Bowers Avenue  
Santa Clara, CA 95051

Attn: MCS-48™ Components

TABLE 1. REGISTER ACCUMULATOR.

R <sub>r</sub>	MOV A,R	MOV R,A	XCH A,R	MOV R, #DATA	INC R	DEC R	DJNZ R	ADD A,R	ADDC A,R	ANL A,R	ORL A,R	XRL A,R
R0	F8	A8	28	B8	18	C8	E8	68	78	58	48	D8
R1	F9	A9	29	B9	19	C9	E9	69	79	59	49	D9
R2	FA	AA	2A	BA	1A	CA	EA	6A	7A	5A	4A	DA
R3	FB	AB	2B	BB	1B	CB	EB	6B	7B	5B	4B	DB
R4	FC	AC	2C	BC	1C	CC	EC	6C	7C	5C	4C	DC
R5	FD	AD	2D	BD	1D	CD	ED	6D	7D	5D	4D	DD
R6	FE	AE	2E	BE	1E	CE	EE	6E	7E	5E	4E	DE
R7	FF	AF	2F	BF	1F	CF	EF	6F	7F	5F	4F	DF

TABLE 2. INPUT/OUTPUT.

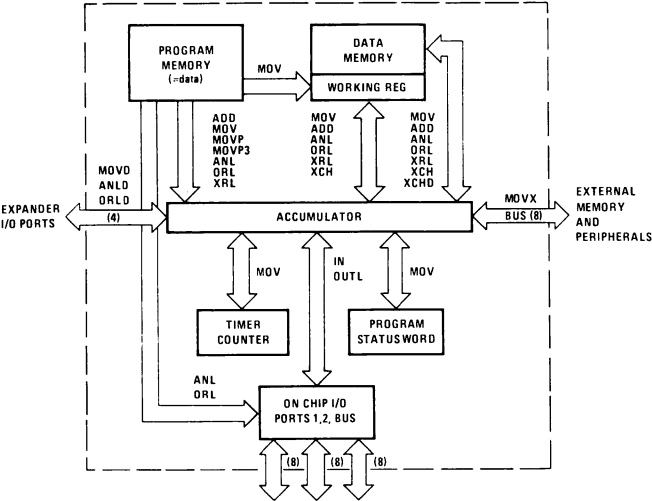
Port	IN	OUT	AND	OR
BUS	08	02	98	88
P1	09	39	99	89
P2	0A	3A	9A	8A
P4	0C	3C	9C	8C
P5	0D	3D	9D	8D
P6	0E	3E	9E	8E
P7	0F	3F	9F	8F

TABLE 3. BRANCH.

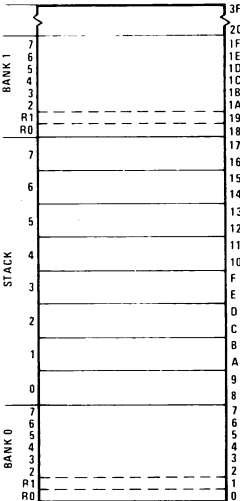
Page	JMP	CALL
0	04	14
1	24	34
2	44	54
3	64	74
4	84	94
5	A4	B4
6	C4	D4
7	E4	F4

Page = 256 bytes

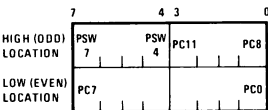
MCS-48™ DATA TRANSFER INSTRUCTIONS



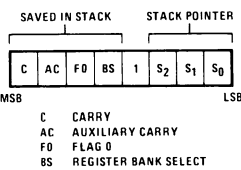
DATA RAM



STACK FORMAT



PROGRAM STATUS WORD (PSW)





INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) 246-7501